



Guida all'hosting

Amazon GameLift Servers



Amazon GameLift Servers: Guida all'hosting

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

I marchi e l'immagine commerciale di Amazon non possono essere utilizzati in relazione a prodotti o servizi che non siano di Amazon, in una qualsiasi modalità che possa causare confusione tra i clienti o in una qualsiasi modalità che denigri o discrediti Amazon. Tutti gli altri marchi non di proprietà di Amazon sono di proprietà delle rispettive aziende, che possono o meno essere associate, collegate o sponsorizzate da Amazon.

Table of Contents

Cos'è Amazon GameLift Servers?	1
Funzionalità principali	1
Come utilizzare Amazon GameLift Servers	4
opzioni di hosting di giochi	5
EC2 gestito	5
Contenitori gestiti	6
Hosting ovunque	7
Hosting ibrido	7
Come funziona	8
Componenti principali	8
Come interagiscono i componenti	16
Esperienza del giocatore	17
Offri un gameplay veloce ai giocatori di tutto il mondo	17
Aiuta i giocatori a partecipare rapidamente alle partite	18
Sviluppa modi creativi per raggruppare i giocatori per le sessioni di gioco	18
Offri connessioni affidabili durante tutto il gioco	19
Migliora l'esperienza dei giocatori sulla base di dati del mondo reale	20
Integra funzionalità di gioco avanzate con altri servizi AWS	20
Hosting di giochi in produzione	21
Monitoraggio dello stato e delle prestazioni dell'hosting	21
Gestione degli aggiornamenti e delle patch dei server di gioco	22
Ottimizzazione delle prestazioni e della scalabilità	22
Risoluzione dei problemi e operazioni in tempo reale	23
Strumenti e integrazione	24
Risorse correlate	24
Termini chiave	24
Nozioni di base	33
Prima di iniziare	33
Scegli il tuo percorso	33
Voglio esplorare Amazon GameLift Servers velocemente	33
Voglio creare una soluzione di hosting personalizzata	34
Percorso di apprendimento consigliato	34
Configura un AWS account	35
Registrati per un Account AWS	36

Crea un utente con accesso amministrativo	37
Imposta le autorizzazioni utente per Amazon GameLift Servers	38
Configura l'accesso programmatico per gli utenti	39
Configura l'accesso programmatico per il tuo gioco	41
Esempi di autorizzazioni IAM	42
Configura un ruolo di servizio IAM	47
Ottieni strumenti di sviluppo	51
Per i server di gioco	51
Per i client di gioco	54
Per i servizi di backend dei giochi	55
Per la gestione delle risorse	55
Per Amazon GameLift Servers Realtime	56
Tutorial: onboarding rapido con il wrapper Amazon GameLift Servers	57
Prerequisiti	58
Panoramica di	58
Passaggio 1: Ottieni e crea il wrapper del server di gioco	58
Passaggio 2: prepara la build del server di gioco	59
Fase 3: Configura il wrapper per la tua flotta	60
Passaggio 4: carica la build del server di gioco	62
Fase 5: Crea la flotta EC2 gestita	62
Passaggio 6: Crea e connettiti a una sessione di gioco	63
Fase 7: Gestisci e monitora la tua flotta	65
Passaggio 8: ridimensiona i server di gioco	65
Risolvi i problemi più comuni	66
Fasi successive	68
Esempi di Amazon GameLift Servers	68
Esempio di server di gioco personalizzato	68
Creazione di una soluzione di hosting di giochi	70
Roadmap di sviluppo	71
Roadmap di sviluppo EC2 gestita	71
Roadmap per lo sviluppo di container gestiti	77
Roadmap di sviluppo ovunque	85
Roadmap di sviluppo dell'hosting ibrido	91
Prepara un gioco per l'hosting	99
Preparati con il plugin del motore di gioco	99
Integra un server di gioco	186

Package di una build del server di gioco	215
Amazon GameLift Serversinterazioni	228
Crea un servizio di backend	234
Autenticazione dei giocatori	235
Integra la funzionalità del client di gioco	235
Backend senza server	241
WebSocketbackend basato	243
Implementa flotte di hosting	245
Caratteristiche della flotta	245
Scegli un'opzione di hosting	246
flotte gestite EC2	248
flotte di container gestite	260
Flotte ovunque	283
Crea una soluzione di hosting ibrida	293
Configura il posizionamento delle sessioni di gioco	296
Caratteristiche della coda	297
Best practice	298
Crea una coda	299
Imposta la notifica degli eventi	303
Personalizza la tua soluzione di hosting di giochi	307
Build di server di gioco	308
Sessioni con i giocatori e matchmaking	321
Posizionamenti delle sessioni di gioco	323
Risorse di hosting	339
Gestione di una soluzione di hosting di giochi	368
Gestisci le risorse di hosting dei giochi	368
Amazon GameLift Serversfunzionalità della console	368
Dashboard della console	369
Compilazioni	370
Flotte	374
Alias	396
Code di sessioni di gioco	399
Preparati per il lancio del gioco	402
Preparazione	402
Preparati per il test	403
Preparati per il lancio	403

Piano per la fase successiva al lancio	404
Gestisci le risorse di hosting dei giochi con CloudFormation	404
Best practice	405
Utilizzo delle CloudFormation pile	406
Aggiornamento delle build	409
DDoSProtezione S	412
Vantaggi principali	412
Come funziona Player Gateway	412
Flusso di traffico	413
Concetti principali	413
Monitoraggio del gateway dei giocatori	415
IPv4 e compatibilità IPv6	415
Abilita il player gateway sulle flotte	415
Prerequisiti	415
Modalità Player Gateway	416
IPv4 e IPv6 compatibilità	416
Integra il gateway del giocatore in un gioco	417
Integrazione con il backend	417
Integrazione con i client	419
Metti alla prova la tua integrazione	421
Best practice	421
Test e risoluzione dei problemi	422
Configurato per lo sviluppo iterativo	422
Crea un ambiente di test basato sul cloud	424
Imposta i test locali	428
Configura i test locali (legacy)	433
Debug dei problemi del parco istanze	441
Problemi di creazione del parco istanze	442
Problemi dei processi del server	444
Problemi di eliminazione del parco istanze	444
Accedi in remoto alle risorse della flotta	445
Connect a flotte di istanze	445
Connect ai container	454
Dimensionamento	462
Per gestire la capacità della flotta nella console	463
Imposta i limiti di capacità di hosting	463

Per impostare i limiti di capacità	464
Impostazione manuale della capacità del parco istanze	465
Sospensione del dimensionamento automatico	465
Per impostare manualmente la capacità del parco istanze	466
Capacità della flotta con scalabilità automatica	467
Dimensionamento automatico basato sugli obiettivi	468
Scalabilità automatica basata su regole	470
Gestisci la scalabilità di una flotta zero To/From	474
Comportamento di scalabilità	475
Comportamento di scalabilità orizzontale	475
To/From Configurazione di Scale Zero	476
Best practice	477
Scalate le flotte di container	478
Monitoraggio	480
Confronto delle metriche	481
Disponibilità delle metriche per fonte	481
Argomenti	484
Monitora con CloudWatch	485
Dimensioni delle metriche	486
Metriche della flotta	487
DDoMetriche di protezione S (player gateway)	504
Parametri coda	506
Parametri di FlexMatch	510
Parametri di FleetIQ	514
Monitoraggio con metriche di telemetria del server	516
Vantaggi delle metriche di telemetria	517
Prima di iniziare	518
Implementazione	519
Metriche disponibili	520
Come funziona	525
Organizzazione e utilizzo del pannello di controllo	526
Scenari di monitoraggio comuni	537
Guida alla risoluzione dei problemi	543
Registrazione di chiamate API	545
Amazon GameLift Serversinformazioni in CloudTrail	546
Informazioni sulle voci dei file di log di Amazon GameLift Servers	547

Registrazione dei messaggi del server	549
EC2 vs Registrazione dei container	550
Registrazione per server personalizzati	552
Sicurezza	556
Protezione dei dati	557
Crittografia dei dati a riposo	558
Crittografia dei dati in transito	559
Riservatezza del traffico Internet	560
Gestione dell'identità e degli accessi	560
Destinatari	561
Autenticazione con identità	561
Gestione dell'accesso tramite policy	562
Funzionamento di Amazon GameLift Servers con IAM	564
Esempi di policy basate su identità	571
Risoluzione dei problemi	576
AWS politiche gestite	578
Registrazione e monitoraggio con Amazon GameLift Servers	580
Convalida della conformità	581
Resilienza	581
Sicurezza dell'infrastruttura	582
Analisi della configurazione e delle vulnerabilità	583
Best practice di sicurezza	584
Mantieni aggiornati gli ambienti di runtime della flotta	584
Proteggi le configurazioni delle porte	586
Risorse di sicurezza aggiuntive	586
Prezzi	587
Modelli tariffari	587
Generare stime dei prezzi	588
Stima dell'hosting gestito	588
Strategie di ottimizzazione dei costi	591
Best practice	592
Costi delle risorse e strumenti di utilizzo	594
Monitoraggio delle prestazioni	594
guide di riferimento	596
API di servizio (AWS SDK)	596
Gestisci le risorse Amazon GameLift Servers di hosting	597

Avvia sessioni di gioco e unisciti ai giocatori	601
Server SDK 5.x	602
Aggiornamenti Server SDK 5.x	603
Esegui la migrazione al server SDK 5.x	604
SDK per server C++	606
SDK per server C#	659
Vai al server SDK	706
SDK per server C++ (Unreal): azioni	737
Server SDK 4 e versioni precedenti	784
Server SDK per C++: azioni	785
Server SDK per C#: azioni	808
Server SDK per Unreal: azioni	830
Eventi di posizionamento delle sessioni di gioco	846
Sintassi degli eventi di posizionamento	846
PlacementFulfilled	847
PlacementCancelled	849
PlacementTimedOut	850
PlacementFailed	851
Versioni delle AMI	852
sedi di assistenza	854
AWS Località supportate	855
Sedi per l'hosting gestito	860
Posizioni per qualsiasi luogo Amazon GameLift Servers	861
Sedi per Amazon GameLift Servers FlexMatch	862
Amazon GameLift Servers in Cina	862
Endpoint di servizio e Service Quotas	863
Limiti delle API di Amazon GameLift Servers	863
Riferimento ai limiti delle API	863
Limitazione a livello di risorsa	872
Limiti delle API SDK di Amazon GameLift Servers	873
UDP ping beacon	878
Come funzionano i ping beacon UDP	878
Casi d'uso comuni dei beacon ping UDP	878
Ottenere endpoint beacon	879
Implementazione della misurazione della latenza	880
Esempi di codice	881

FAQs sui protocolli supportati	889
Cosa sono i protocolli AWS CBOR JSON e Smithy RPC v2 e in che cosa differiscono?	889
Come posso iniziare a usare il protocollo CBOR Smithy RPC v2?	889
Quali sono i rischi dell'abilitazione del protocollo Smithy RPC v2 CBOR?	890
Cosa succede se la mia soluzione non supporta Smithy RPC v2 CBOR?	890
Quali AWS regioni supportano il protocollo Smithy RPC v2 CBOR?	890
Quali miglioramenti della latenza posso aspettarmi eseguendo l'aggiornamento a versioni SDK che supportano Smithy RPC v2 CBOR?	890
Il protocollo JSON sarà obsoleto? AWS	891
Dove posso trovare ulteriori informazioni su Smithy RPC v2 CBOR?	891
Note di rilascio e versioni SDK	892
Versioni SDK	892
Note di rilascio	907
.....	cmlxi

Cos'è Amazon GameLift Servers?

Amazon GameLift Servers Utilizzalo per distribuire, gestire e scalare server dedicati a basso costo nel cloud per giochi multiplayer basati su sessioni. Basato su un'infrastruttura informatica AWS globale, Amazon GameLift Servers aiuta a fornire server di gioco ad alte prestazioni e alta affidabilità, scalando dinamicamente l'utilizzo delle risorse per soddisfare la domanda dei giocatori in tutto il mondo.

Funzionalità principali

Copertura globale e bassa latenza

- Implementa l'hosting nelle AWS Regioni e nelle Local Zones di tutto il mondo per portare i tuoi giochi a nuovi giocatori. Consulta l'elenco completo delle [località supportate da Amazon GameLift Servers](#).
- Offri la latenza più bassa possibile e un'esperienza di gioco eccezionale, ovunque si trovino i giocatori.
- Utilizza code di posizionamento in più sedi per collocare dinamicamente i giocatori in sessioni di gioco con esperienze di gioco ottimali.

Scalabilità automatica per flotte gestite

- Ottimizza la capacità della flotta per ridurre i costi mantenendo al contempo la qualità del servizio.
- Scala la capacità in base alla domanda dei giocatori utilizzando la scalabilità automatica basata sul target
- Mantieni il buffer di capacità per gestire afflussi improvvisi di giocatori.

Opzioni di hosting flessibili

Vedi maggiori dettagli su [Amazon GameLift Servers opzioni di hosting di giochi](#).

- Hosting di giochi nel cloud, gestito da Amazon GameLift Servers
 - Ospita server di gioco sull'infrastruttura Amazon Elastic Compute Cloud (Amazon EC2). Scegli tra tipi di risorse tra cui istanze Graviton basate su ARM ottimizzate per il calcolo (famiglia C), ottimizzate per la memoria (famiglia R) e convenienti.

- Personalizza il modo in cui Amazon GameLift Servers gestisce tutti gli aspetti della tua soluzione di hosting di giochi, dall'implementazione dei server di gioco all'avvio delle sessioni di gioco, fino all'abbinamento dei giocatori e al loro coinvolgimento nelle partite.
- Utilizza flotte con più sedi per semplificare l'implementazione globale dei server di gioco.
- Sfrutta i vantaggi dell'hosting a basso costo con le istanze Spot di [Amazon EC2](#).
- Implementa server di gioco Windows o Linux con hosting EC2 gestito.
- Implementa contenitori basati su Linux con il tuo server di gioco con hosting gestito di container.
- Hosting di giochi gestito da te, con Anywhere Amazon GameLift Servers
 - Sfrutta le funzionalità di posizionamento e gestione delle sessioni di Amazon GameLift Servers gioco.
 - Gestisci autonomamente le implementazioni e la scalabilità dei server di gioco sul tuo hardware, sull'infrastruttura locale o su altri provider cloud.
- Hosting ibrido di giochi
 - Crea una soluzione di hosting di giochi che utilizzi una combinazione di flotte Amazon GameLift Servers gestite e flotte Anywhere per utilizzare le soluzioni esistenti o supportare la migrazione.

Gestione delle sessioni di gioco

- Lascia che Amazon GameLift Servers gestisca i processi del server di gioco per te, monitora lo stato e la disponibilità del server di gioco e organizza nuove sessioni di gioco su richiesta.
- Usa le funzioni di posizionamento delle sessioni di gioco per aiutare i giocatori a trovare e partecipare a sessioni di gioco che offriranno la migliore esperienza di gioco possibile. Affidati al Amazon GameLift Servers processo decisionale o personalizza in base a criteri come il costo di hosting, la latenza dei giocatori e la posizione geografica.
- Crea code di posizionamento delle sessioni di gioco per elaborare in modo efficiente le nuove richieste di sessioni di gioco. Usa le funzionalità per stabilire un equilibrio tra qualità e velocità di posizionamento che soddisfi le tue aspettative di gioco e dei giocatori.
- Sfrutta le funzionalità opzionali della sessione giocatore per prenotare gli slot nelle sessioni di gioco, convalidare i giocatori al momento della connessione e monitorare la disponibilità degli slot. Cerca o filtra le sessioni di gioco esistenti per trovare gli slot disponibili.

Matchmaking avanzato con FlexMatch

- Aggiungi un matchmaking robusto e completamente integrato con la tua soluzione di Amazon GameLift Servers hosting.
- Crea regole di matchmaking personalizzate in base agli attributi e alle preferenze del giocatore, come il livello di abilità o la scelta del personaggio, e usa una logica complessa secondo necessità.
- Imposta partite a squadre per un gameplay competitivo o cooperativo. Equilibra i giocatori in and/or base alla composizione delle partite di squadra.
- Ottimizza l'algoritmo di matchmaking per bilanciare la qualità delle partite e i tempi di attesa.
- Supporta partite da 2 a 200 giocatori simultanei.
- Supporta il match backfilling per riempire gli slot disponibili per i giocatori e mantenere piene le sessioni di gioco.

Monitoraggio completo

- Ottieni metriche in tempo reale sull'utilizzo e le prestazioni per le sessioni di gioco e l'attività dei giocatori, oltre a ospitare le prestazioni e lo stato delle flotte gestite.
- Effettua l'integrazione con Amazon CloudWatch per avvisi e dashboard.
- Accedi ai log del server e al monitoraggio degli eventi per la risoluzione dei problemi.
- Analizza i dati sulle prestazioni e sull'utilizzo per ottimizzare la scalabilità (flotte gestite) o l'allocazione delle risorse (flotte autogestite).

Personalizzazione ed estensibilità

- Personalizza la logica di posizionamento delle sessioni di gioco, inclusa l'assegnazione di priorità per latenza, costo e posizione.
- Per l'hosting gestito, scegli le risorse di hosting dei giochi e fornisci istruzioni per l'avvio e l'esecuzione dei server di gioco su ciascuna istanza. Per i container gestiti, ottimizza l'allocazione delle risorse tra i container.
- Definisci le tue regole di matchmaking.
- Implementa strategie personalizzate per la scalabilità della capacità.
- Aggiungi il codice del server di gioco da integrare con i tuoi AWS servizi esistenti, come database o strumenti di archiviazione dei contenuti.

- Crea un servizio di backend con funzionalità personalizzate per gestire le richieste di iscrizione dei giocatori (come i party dei giocatori), gestire l'autenticazione o aggiungere logica e persistenza personalizzate.

Integrazione con l'ecosistema AWS

- Utilizza servizi come Amazon DynamoDB, Amazon Simple Storage Service e Amazon Aurora DSQL per la persistenza dello stato di gioco.
- Implementa Amazon Cognito per l'autenticazione dei giocatori.
- Elabora l'analisi dei giochi con Amazon Kinesis e Amazon S3.
- Aggiungi chat vocale con Amazon Chime SDK.
- Crea funzionalità di gioco personalizzate utilizzando AWS Lambda Amazon API Gateway.

Come utilizzare Amazon GameLift Servers

Utilizzare questi strumenti per l'utilizzo di Amazon GameLift Servers.

AWS CLI

Usa AWS Command Line Interface (AWS CLI) per effettuare chiamate all' AWS SDK, inclusa l'API del servizio per Amazon GameLift Servers. Vedi [Guida introduttiva alla AWS CLI](#) Guida per l'AWS Command Line Interface utente.

Console Amazon GameLift Servers

Utilizza il form [Console di gestione AWS Amazon GameLift Servers](#) per configurare le risorse, gestire le distribuzioni dei server di gioco e tenere traccia delle metriche di prestazioni e utilizzo. La Amazon GameLift Servers console è un'alternativa alla GUI alla gestione delle risorse a livello di codice o con AWS CLI

Amazon GameLift Servers SDKs

Amazon GameLift Servers SDKs Contengono le librerie necessarie per stabilire la comunicazione tra i client di gioco, i server di gioco e i servizi di gioco e il servizio. Amazon GameLift Servers Per ulteriori informazioni, consulta [Ottieni strumenti Amazon GameLift Servers di sviluppo](#).

AWS CloudFormation

Utilizzali AWS CloudFormation per modellare, fornire e gestire AWS le risorse per la tua soluzione di hosting di giochi trattando l'infrastruttura come codice. Crea modelli che descrivono le risorse

e CloudFormation automatizzano le attività di configurazione e distribuzione delle risorse nelle posizioni specificate.

Amazon GameLift Servers opzioni di hosting di giochi

Amazon GameLift Servers offre una gamma di opzioni per ospitare server di gioco per giochi multiplayer basati su sessioni. Sia che tu voglia configurare un hosting basato su cloud gestito da Amazon GameLift Servers o incorporare risorse di hosting che già possiedi, puoi collaborare con Amazon GameLift Servers per creare la soluzione di hosting di cui hai bisogno per i tuoi giocatori. Per maggiori dettagli sull'aspetto di una soluzione di hosting di giochi, consulta. [Come Amazon GameLift Servers funziona l'hosting con](#)

Argomenti

- [EC2 gestito](#)
- [Contenitori gestiti](#)
- [Hosting ovunque](#)
- [Hosting ibrido](#)

EC2 gestito

Affida il lavoro di gestione dei server di gioco di produzione Amazon GameLift Servers con l'hosting EC2 gestito. Sfrutta le ottimizzazioni per i server di gioco multiplayer affidandoti comunque alle alte prestazioni e all'affidabilità di Amazon Elastic Compute Cloud (Amazon EC2) e dell'infrastruttura di elaborazione globale. AWS

[Inizia a creare una soluzione di hosting EC2 gestita Amazon GameLift Servers](#)

Caratteristiche

- Ospita server di gioco multiplayer che funzionano su sistemi operativi Amazon Linux o Windows Server.
- Amazon GameLift Servers Utilizzalo per distribuire e gestire i tuoi server di gioco personalizzati su risorse Cloud AWS di hosting. Scegli il tipo di hardware e dove installarlo e configura dettagli aggiuntivi solo dove necessario.
- Configura il gestore dei processi di runtime per gestire automaticamente i processi del server di gioco in base alle esigenze di ospitare le sessioni di gioco.

- Imposta un monitoraggio personalizzato dello stato di salute per Amazon GameLift Servers rilevare e risolvere i server di gioco con prestazioni scadenti.
- Sfrutta il monitoraggio delle Amazon GameLift Servers prestazioni. Tieni traccia di parametri come le prestazioni dell'hardware e i cicli di vita dei processi del server. Puoi anche scaricare e archiviare i registri delle sessioni di gioco.
- Utilizzalo Amazon GameLift Servers per gestire l'accesso tra i processi del server di gioco e altre AWS risorse.
- Per l'hosting di produzione, automatizza le implementazioni e la gestione dell'hosting di giochi con AWS CloudFormation modelli e. AWS Cloud Development Kit (AWS CDK) Sfrutta strumenti e servizi di integrazione continua e distribuzione continua (CI/CD) come. AWS CodePipeline

Contenitori gestiti

Amazon GameLift Servers offre una soluzione di hosting cloud completa per server di gioco containerizzati, in modo da poter sfruttare i vantaggi principali dell'utilizzo dei container, come portabilità, agilità e tolleranza agli errori. Analogamente all'hosting EC2 gestito, l'hosting gestito di container distribuisce ed esegue i contenitori su istanze Amazon EC2.

[Inizia a sviluppare una soluzione di hosting Amazon GameLift Servers gestito per il tuo server di gioco containerizzato.](#)

Caratteristiche

- Sviluppa un'architettura personalizzata con contenitori leggeri per eseguire la build del server di gioco, le dipendenze e il software ausiliario.
- Usa gli strumenti Docker per creare un'immagine di contenitore basata su Linux. Archivia le immagini in un repository Amazon Elastic Container Registry (Amazon ECR) per la distribuzione.
- Utilizzalo Amazon GameLift Servers per distribuire e gestire server di gioco personalizzati e risorse di hosting. Cloud AWS Scegli il tipo di hardware e dove installarlo e configura dettagli aggiuntivi solo dove necessario.
- Gestisci il ciclo di vita della flotta di hosting con strumenti per modellare le versioni dei server di gioco. Implementa gli aggiornamenti dei server di gioco e altre modifiche alla configurazione secondo necessità.
- Tieni traccia delle prestazioni di hosting dei giochi con metriche prestazionali specifiche del contenitore. Monitora lo stato delle risorse della tua flotta utilizzando metriche hardware.

- Gestisci le risorse della flotta di container utilizzando AWS CloudFormation modelli per Amazon GameLift Servers.

Hosting ovunque

Con l'hosting Anywhere, puoi sfruttare le funzionalità di gestione delle sessioni di Amazon GameLift Servers gioco, tra cui matchmaking e metriche delle sessioni di gioco, per ospitare server di gioco ovunque tu voglia. Gestisci autonomamente le implementazioni dei server di gioco, il monitoraggio dello stato dei server di gioco e la scalabilità della capacità.

Note

Anywherelle flotte sono particolarmente utili come ambienti di test per lo sviluppo rapido e iterativo di giochi. Configura una Anywhere flotta per la tua workstation locale o per una risorsa di hosting basata su cloud. Per ulteriori informazioni sui test con Anywhere, consulta. [Configura i test locali con Amazon GameLift Servers Anywhere](#)

[Inizia a sviluppare una soluzione di hosting Amazon GameLift Servers Anywhere per il tuo gioco.](#)

Caratteristiche

- Implementa i server di gioco sul tuo hardware, sull'infrastruttura locale o su altri provider cloud. Sfrutta l'hardware disponibile vicino ai tuoi giocatori, ovunque.
- Usa le funzionalità di gestione delle Amazon GameLift Servers sessioni per monitorare la disponibilità dei server di gioco, avviare nuove sessioni di gioco e unire i giocatori agli slot delle sessioni di gioco.

Hosting ibrido

Per l'hosting di produzione, potresti scegliere un approccio ibrido, con Anywhere flotte per le risorse di hosting che fornisci e gestisci e flotte Amazon GameLift Servers gestite che si aggiungono dove necessario. Una soluzione ibrida utilizza gli stessi processi per avviare sessioni di gioco e connettere i giocatori per offrire un'esperienza di gioco senza interruzioni. Puoi monitorare e raccogliere i parametri di utilizzo per tutti gli hosting, indipendentemente dal tipo di risorsa.

Scenari comuni in cui una soluzione ibrida ha senso:

- Espandi la tua soluzione di hosting a Cloud AWS. Completa le funzionalità della tua soluzione di hosting esistente (hardware locale o altro hosting basato sul cloud) aggiungendo l'Amazon GameLift Servers hosting gestito. Con l'hosting gestito, puoi aumentare la capacità di hosting o aggiungere capacità «burst» per scalare rapidamente e pagare solo le risorse quando ne hai bisogno. Puoi anche sfruttare l'impronta globale del Amazon GameLift Servers servizio per raggiungere più giocatori in tutto il mondo e offrire l'esperienza multiplayer a bassa latenza che si aspettano.
- Preparati per la migrazione all'hosting basato sul cloud. Se stai considerando o hai intenzione di migrare a Cloud AWS, una soluzione di hosting ibrido è un modo valido per effettuare la transizione il più gradualmente necessario. Ti consente di creare la soluzione di cui hai bisogno in questo momento mentre ti prepari per le prospettive future.
- Latenza inferiore per i giocatori in località diverse da quelle servite da. Amazon GameLift Servers. Se utilizzi già l'hosting Amazon GameLift Servers gestito, potresti dover supportare i giocatori in situazioni insolite. Ad esempio, potresti voler raggiungere meglio i giocatori che si trovano in località insolitamente remote o supportare giocatori con esigenze di accesso speciali. Aggiungi sedi di hosting personalizzate e usa Amazon GameLift Servers Anywhere per supportare tali sedi insieme alle tue risorse di hosting gestite.
-

[Inizia a sviluppare una soluzione di hosting Amazon GameLift Servers ibrida per il tuo gioco.](#)

Come Amazon GameLift Servers funziona l'hosting con

Amazon GameLift Servers è un servizio che fornisce server dedicati a basso costo, fornitura dell'infrastruttura, scalabilità e gestione delle sessioni per i giochi multiplayer basati su sessioni. Il servizio offre strumenti e funzionalità flessibili che puoi personalizzare per i tuoi giochi o utilizzare in collaborazione con i tuoi sistemi. Sfrutta l'hosting Amazon GameLift Servers gestito in modo da poterti concentrare sulla creazione di esperienze di gioco anziché sulla gestione delle operazioni del server.

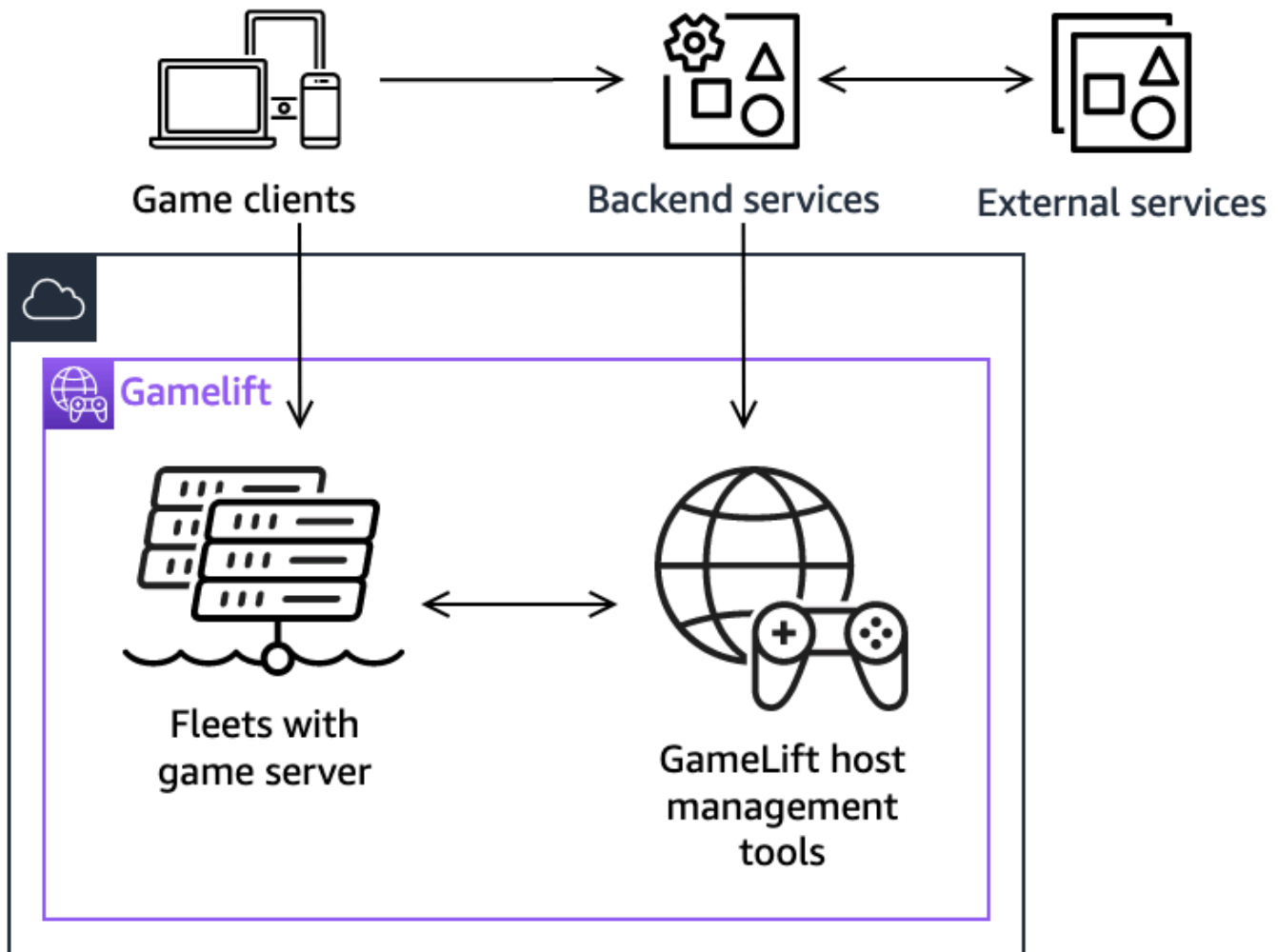
Questo argomento descrive i componenti principali di una soluzione di Amazon GameLift Servers hosting completa e il modo in cui interagiscono per fornire sessioni di gioco multiplayer ai giocatori.

Componenti principali di una soluzione di hosting di giochi

Una soluzione di Amazon GameLift Servers hosting completa è composta da diversi componenti essenziali che interagiscono per offrire esperienze di gioco multiplayer. Alcuni di questi componenti

sono creati e gestiti dall'utente, mentre altri sono forniti Amazon GameLift Servers e gestiti in base alle scelte di configurazione dell'utente. Comprendere come interagiscono questi componenti e i rispettivi ruoli è fondamentale per lo sviluppo di una soluzione di hosting di giochi efficace. I componenti includono:

- Il client di gioco funziona sui dispositivi dei giocatori.
- Il servizio di backend consente ai client di gioco di comunicare con loro Amazon GameLift Servers per ottenere informazioni sulla sessione di gioco e partecipare ai giochi.
- Il software del server di gioco funziona su risorse di hosting dei giochi e ospita sessioni di gioco per un gruppo di giocatori.
- Il sistema di posizionamento avvia le sessioni di gioco e abbina i giocatori alle partite.
- Le flotte di hosting di giochi forniscono server di gioco in una o più località geografiche.
- Il sistema di gestione dell'hosting di giochi monitora lo stato di hosting dei giochi e gestisce la capacità.



Client di gioco

Il client di gioco è il software di gioco che gira sul dispositivo di un giocatore. Avvia il posizionamento del giocatore in una sessione di gioco comunicando con un servizio di backend e si connette direttamente a un server di gioco per partecipare al gioco.

Funzioni chiave per l'hosting di giochi

- Invia richieste di iscrizione al servizio di backend. Includi i dati pertinenti del giocatore e della sessione di gioco come parte della richiesta di iscrizione.
- Raccogli i dati di latenza per il client di gioco e includi tali informazioni nella richiesta di iscrizione.
- Ricevi le informazioni sulla connessione della sessione di gioco dal servizio di backend e usale per connetterti a un server di gioco.

- Gestisci scenari di connessione alla sessione come la verifica del giocatore, l'accettazione di una partita Amazon GameLift ServersFlexMatch, le interruzioni della sessione o le disconnessioni.

Chi lo costruisce

Aggiungete la funzionalità di hosting dei giochi al software del client di gioco e impostate la comunicazione con il servizio di backend.

Servizio di backend

Il servizio di backend è un livello di coordinamento tra i client di gioco e il Amazon GameLift Servers servizio. Controlla tutte le comunicazioni con il servizio per effettuare richieste di posizionamenti nelle sessioni di gioco e recuperare informazioni sulla sessione di gioco e sulla sessione dei giocatori. L'uso di un servizio di backend è una best practice che mantiene una comunicazione sicura con il servizio per conto dei client di gioco ed evita di dover condividere AWS credenziali e autorizzazioni sensibili con i client di gioco.

Il tuo servizio di backend implementa una strategia di raggruppamento dei giocatori per le tue sessioni di gioco. Hai molta flessibilità nel modo in cui i giocatori finiscono in una sessione di gioco. Potresti configurare un browser per le sessioni di gioco per i giocatori o supportare gruppi definiti dai giocatori. Oppure puoi raggruppare le richieste di sessioni di gioco man mano che arrivano, formare gruppi di giocatori (possibilmente con un preordinamento) e iniziare una sessione di gioco per ogni gruppo. Con il FlexMatch matchmaking, puoi creare partite tra giocatori e completare le partite esistenti.

Funzioni chiave per l'hosting di giochi

- Autentica le comunicazioni dai client di gioco.
- Effettua richieste Amazon GameLift Servers per iniziare nuove sessioni di gioco, ottenere informazioni sulla sessione di gioco o partecipare a sessioni di gioco esistenti. Se necessario, includi i dati pertinenti della sessione di gioco e dei giocatori nelle richieste.
- Facoltativamente, crea richieste per le sessioni dei giocatori per monitorare più da vicino la disponibilità delle sessioni di gioco. Usa le sessioni dei giocatori per prenotare gli slot delle sessioni di gioco, convalidare i giocatori quando si connettono e tenere traccia delle disconnessioni dei giocatori.
- Rispondi ai client di gioco con i dettagli della connessione alla sessione di gioco e altre informazioni come richiesto.

Chi lo costruisce

Crei il servizio di backend in un ambiente gestito da te per supportare i tuoi requisiti di gioco e implementare il modo in cui i giocatori entrano nei giochi. Integra l' AWS SDK per effettuare chiamate all'API del Amazon GameLift Servers servizio. Opzionalmente, crea il tuo servizio di backend AWS utilizzando servizi come AWS Lambda Amazon Simple Storage Service (Amazon S3) e Amazon Cognito.

Server di gioco

Il server di gioco è un software server personalizzato che gestisce lo stato del gioco, elabora le azioni dei giocatori e sincronizza il gameplay tra più giocatori connessi. Il server di gioco mantiene la comunicazione con il Amazon GameLift Servers servizio per gestire l'hosting delle sessioni di gioco.

Funzioni chiave per l'hosting di giochi

- Comunica con Amazon GameLift Servers a:
 - segnala lo stato (pronto per ospitare sessioni, pronto ad accettare giocatori, stato di salute).
 - rispondere alle chiamate di servizio (inizio o fine delle sessioni di gioco).
- Gestisci il ciclo di vita della sessione di gioco per ospitare una sessione di gioco alla volta per processo.
- Coordinati con altri AWS servizi per funzionalità aggiuntive.
- Facoltativamente, convalida le connessioni di nuovi giocatori.

Chi lo costruisce

Crei il software del tuo server di gioco. Integri l'SDK del server Amazon GameLift Servers e aggiungi funzionalità per stabilire una connessione con il servizio e supportare la gestione delle sessioni di gioco.

Flotta di hosting di giochi

La flotta di hosting è una raccolta di risorse informatiche che gestiscono i tuoi server di gioco. Le risorse della flotta possono essere distribuite su più aree geografiche per offrire ai giocatori un gameplay a bassa latenza ovunque si trovino. Ogni risorsa della flotta esegue uno o più processi del server di gioco, che comunicano direttamente con. Amazon GameLift Servers Ogni processo del server di gioco può ospitare una sessione di gioco alla volta.

Le caratteristiche e le funzionalità della flotta variano in base al tipo di hosting della flotta. Le flotte gestite distribuiscono risorse Cloud AWS e sono gestite da Amazon GameLift Servers. Le flotte Anywhere sono risorse di elaborazione fornite dal cliente e gestite all'esterno. Amazon GameLift Servers

Funzioni chiave per l'hosting di giochi

- Fornisci risorse di hosting per server di gioco.
 - Le flotte gestite distribuiscono istanze Amazon EC2 basate sul cloud con un'ampia gamma di impostazioni di configurazione. Una configurazione del parco istanze determina la potenza di calcolo di ciascuna istanza del parco istanze, la posizione fisica delle istanze e altri dettagli. Le flotte gestite aggiungono o rimuovono istanze EC2 in risposta alla scalabilità della capacità.
 - Ovunque, le implementazioni della flotta sono autogestite. La flotta può essere composta da hardware fisico o altre risorse cloud ed essere configurata in base alle esigenze.
- Installa l'ambiente di runtime e il software del server di gioco.
 - Le istanze gestite del parco istanze Amazon EC2 vengono distribuite con la build del server di gioco e un'Amazon Machine Image (AMI) con un ambiente di runtime compatibile.
 - Le istanze gestite di flotte di container vengono distribuite con la build del server di gioco e un'AMI ottimizzata per container con strumenti Docker e altri componenti per funzionare con Amazon ECS.
 - Ovunque le flotte vengano distribuite con il software e il sistema operativo del server di gioco prescelti.
- Gestisci il ciclo di vita dei processi del server di gioco. Applica istruzioni di runtime preconfigurate per avviare e interrompere i processi su ogni risorsa del parco macchine.

Chi lo costruisce

Dipende dal tipo di hosting della flotta:

- Flotte Amazon EC2 gestite: carichi la build del tuo server di gioco su Amazon GameLift Servers, che la memorizza per la distribuzione su flotte di istanze. Amazon GameLift Servers fornisce l'infrastruttura della flotta. Cloud AWS. Tu configuri la flotta e le spieghi come far funzionare i server di gioco su ogni istanza della flotta.
- Flotte di container gestite: raccogli le istruzioni di build e runtime del server di gioco in un'immagine del contenitore da archiviare in Amazon ECR per la distribuzione. Amazon GameLift

Servers fornisce l'infrastruttura della flotta. Cloud AWS Fornisci un'architettura di container e configuri la flotta per ospitare i tuoi container.

- Flotte ovunque: esegui il provisioning di tutta l'infrastruttura e gestisci l'implementazione del software server. Crea flotte Anywhere a cui connettere le tue risorse di hosting di giochi attive.
- Amazon GameLift Servers

Sistema di posizionamento delle sessioni di gioco

Il sistema di posizionamento delle sessioni di gioco individua i server di gioco disponibili per ospitare nuove sessioni di gioco. Il sistema utilizza informazioni in tempo reale sulla disponibilità dei server di gioco per prendere decisioni di posizionamento ottimali.

NelAmazon GameLift Servers, il meccanismo principale di posizionamento delle sessioni di gioco è la coda. Una coda di sessioni di gioco utilizza algoritmi, che puoi configurare, per posizionare le sessioni di gioco e ottenere il miglior risultato possibile. Puoi dare priorità ai posizionamenti in base a fattori quali il costo di hosting più basso e la latenza dei giocatori più bassa, e puoi configurare una coda per la ricerca in più località geografiche. In alternativa alle code, puoi designare una flotta specifica per ospitare le tue sessioni di gioco.

Funzioni chiave per l'hosting dei giochi

- Elabora le richieste di posizionamento delle sessioni di gioco ricevute dal servizio di backend.
- Effettua posizionamenti sulla base di informazioni in tempo reale sulla disponibilità delle risorse di hosting.
- Usa i dati sulla latenza dei giocatori e altri dati per dare priorità alle opzioni di posizionamento (solo code)
- Richiedi ai processi del server di gioco di iniziare nuove sessioni di gioco.
- Aggiorna le informazioni di connessione alla sessione di gioco dopo che la sessione di gioco è pronta per accettare giocatori.
- Opzionalmente, configura il FlexMatch matchmaking per creare partite tra giocatori e richiedere il posizionamento delle sessioni di gioco per le partite.

Chi lo costruisce

Amazon GameLift Serversfornisce il sistema di collocamento. Puoi configurare il comportamento di posizionamento attraverso il modo in cui il tuo servizio di backend effettua le richieste di

posizionamento e, facoltativamente, impostando le code delle sessioni di gioco e i matchmaker. FlexMatch Usa la Amazon GameLift Servers console, l' AWS SDK o il AWS CLI per creare e configurare code e matchmaker per il tuo gioco.

Sistema di gestione dell'hosting dei giochi

Il sistema di gestione dell'hosting di giochi è la spina dorsale operativa che coordina e monitora tutti gli aspetti della vostra soluzione di hosting di giochi. Questo sistema fornisce l'intelligenza e l'automazione che rendono Amazon GameLift Servers possibili le funzionalità di posizionamento e scalabilità.

Funzioni chiave per l'hosting di giochi

- Tieni traccia dello stato e della disponibilità in tempo reale dei processi dei server di gioco, delle sessioni di gioco e delle sessioni dei giocatori in tutte le flotte per supportare il posizionamento delle sessioni di gioco e il ridimensionamento automatico della capacità.
- Monitora lo stato e le prestazioni della flotta.
- Raccogli e analizza le metriche delle attività di hosting di giochi.
- Fornisci strumenti di scalabilità della capacità, tra cui il ridimensionamento automatico in base alla domanda dei giocatori e all'utilizzo della flotta.
- Gestisci gli aggiornamenti del software del server di gioco e le istruzioni di runtime.

Chi lo costruisce

- Flotte gestite: Amazon GameLift Servers fornisce il monitoraggio della disponibilità, metriche sull'attività di hosting dei giochi e sulle prestazioni dell'hardware e strumenti di scalabilità della capacità. Utilizzi AWS gli strumenti disponibili per gestire gli aggiornamenti software dei server di gioco, modificare le istruzioni di runtime e personalizzare l'uso delle metriche (come la configurazione CloudWatch per il monitoraggio). È possibile configurare una politica di ridimensionamento personalizzata e modificarla in base alle esigenze.
- Anywhere fleets: Amazon GameLift Servers fornisce il monitoraggio della disponibilità e le metriche delle attività di hosting dei giochi. Gestisci le modifiche alla configurazione della flotta, inclusi gli aggiornamenti del software del server di gioco e le istruzioni di runtime. Crei sistemi per monitorare le prestazioni della flotta e gestire la scalabilità della capacità della flotta.

Come collaborano i componenti

Quando viene distribuito un computer per l'hosting di giochi

- Software del server di gioco installato: il computer viene installato con un ambiente di runtime e la build del server di gioco
- Avvio del server di gioco: sul computer viene avviata almeno un'istanza dell'eseguibile del server di gioco, con parametri di avvio opzionali.
- Il server di gioco si connette a Amazon GameLift Servers: come parte delle sue azioni di avvio, il processo del server di gioco chiama l'SDK del server per inizializzare una connessione al servizio.
- I report del server di gioco sono pronti per ospitare una sessione di gioco: il processo del server di gioco completa le azioni di avvio, quindi chiama l'SDK del server per segnalare la disponibilità. Inizia a segnalare lo stato di salute in base alla sua configurazione.
- Amazon GameLift Servers monitora la disponibilità: il servizio registra la disponibilità del processo del server di gioco al posizionamento della sessione di gioco. Inoltre, aggiorna le metriche relative a flotte, istanze e sessioni di gioco per tenere traccia dell'utilizzo e della capacità.
- Il server di gioco attende l'assegnazione della sessione di gioco: il processo del server di gioco mantiene lo stato di inattività mentre attende la richiesta di avvio di una sessione di gioco.

Quando un giocatore desidera partecipare a una partita

- Il giocatore inizia una partita: un giocatore avvia il client di gioco, si autentica con il servizio di backend e chiede di partecipare a una partita.
- Il servizio di backend richiede una sessione di gioco: il servizio di backend chiama Amazon GameLift Servers per trovare o creare una sessione di gioco adatta, in base alla sua strategia di raggruppamento dei giocatori. La richiesta potrebbe includere dati del giocatore o del gioco da utilizzare nella sessione di gioco.
- Amazon GameLift Servers colloca la sessione di gioco: quando si avvia una nuova sessione di gioco, il sistema di posizionamento identifica una posizione ottimale per ospitare la sessione e seleziona un processo del server di gioco disponibile. Il servizio richiede al processo selezionato di iniziare una nuova sessione di gioco e trasmette tutti i dati del giocatore o del gioco. Se il FlexMatch matchmaking è in uso, il matchmaker crea prima una partita, poi richiede il posizionamento per la partita.

- Il server di gioco avvia la sessione di gioco: Il server di gioco avvia i passaggi per avviare una sessione di gioco. Al termine, segnala Amazon GameLift Servers che è pronto ad accettare le connessioni dei giocatori.
- Informazioni sulla connessione fornite al client di gioco: dopo che il server di gioco ha aggiornato il suo stato, Amazon GameLift Servers fornisce le informazioni sulla connessione alla sessione di gioco. Il servizio di backend riceve queste informazioni e le invia al client di gioco.
- Il giocatore si connette alla sessione di gioco: il client di gioco utilizza le informazioni di connessione per connettersi direttamente al server di gioco e iniziare il gioco.
- Amazon GameLift Servers monitora lo stato della sessione di gioco: il processo del server di gioco segnala lo stato di salute, lo stato della connessione del giocatore opzionale e lo stato della sessione di gioco per tenere traccia della disponibilità delle sessioni di gioco in corso.
- Il processo del server di gioco si spegne: il processo del server di gioco termina la sessione di gioco, segnala lo stato e quindi si spegne automaticamente.

Amazon GameLift Serverse l'esperienza del giocatore

I giocatori si aspettano esperienze di gioco multiplayer veloci, corrette e affidabili. Amazon GameLift Servers offre queste esperienze fornendo agli sviluppatori di giochi strumenti che migliorano direttamente il modo in cui i giocatori si connettono e interagiscono con i giochi multiplayer. Questo argomento descrive i principali vantaggi offerti dalle Amazon GameLift Servers funzionalità per i giocatori.

Offri un gameplay veloce ai giocatori di tutto il mondo

I giocatori vogliono connettersi ai server vicini con un ritardo minimo. Amazon GameLift Servers garantisce esperienze di connessione ottimali attraverso un posizionamento intelligente e un'infrastruttura globale:

- I giocatori ottengono prestazioni costanti a livello globale: organizza partite nelle AWS Regioni e nelle Local Zones disponibili a livello globale per ridurre al minimo la distanza tra i giocatori e i server di gioco (vedi [Amazon GameLift Servers sedi di assistenza](#)). Puoi scegliere tra un'ampia gamma di tipi di EC2 istanze Amazon in queste località per trovare le migliori risorse di elaborazione per ospitare i tuoi carichi di lavoro di gioco.
- I giocatori si connettono ai migliori server disponibili: Amazon GameLift Servers inseriscono i giocatori in sessioni di gioco con la latenza più bassa possibile raccogliendo i dati della rete dei client di gioco e cercando i server disponibili in più località. Utilizza i beacon di ping UDP per

raccogliere dati di latenza accurati. Configura le politiche di latenza per bilanciare la qualità della connessione con tempi di attesa accettabili per i giocatori.

- I giocatori di diverse regioni possono giocare insieme: i posizionamenti delle sessioni di gioco possono bilanciare le sessioni di gioco con giocatori provenienti da pool di giocatori regionali mantenendo una latenza accettabile. I giocatori delle regioni meno popolate possono trovare le partite senza sacrificare la qualità della connessione.

Aiuta i giocatori a partecipare rapidamente alle partite

I giocatori vogliono un accesso rapido ai giochi indipendentemente dalla domanda. Amazon GameLift Servers offre una gamma di strumenti di scalabilità della capacità che aiutano a garantire che i giocatori possano trovare i giochi durante i periodi di punta e di silenzio:

- I giocatori possono trovare facilmente i server disponibili: con gli strumenti di ridimensionamento automatico, puoi assicurarti che le risorse di hosting siano pronte quando sono necessarie. Target Tracking gestisce un buffer di capacità flessibile per anticipare i modelli di domanda dei giocatori per il tuo gioco e scarlo in modo proattivo.
- I giocatori ottengono prestazioni fluide durante gli eventi speciali: combina l'auto-scaling con tecniche di ridimensionamento manuale per gestire afflussi improvvisi dopo eventi pianificati, come aggiornamenti di gioco, campagne di marketing o tornei.
- I giocatori non devono aspettare l'avvio dei server: su ogni risorsa di hosting di giochi, i server di gioco sono preriscaldati in modo da essere sempre pronti a ospitare nuove sessioni non appena i giocatori si uniscono.

Sviluppa modi creativi per raggruppare i giocatori per le sessioni di gioco

I giocatori vogliono competere o collaborare con gli altri in modo equo ed equilibrato. Amazon GameLift Servers offre una serie di opzioni per raggruppare i giocatori in sessioni di gioco:

- I giocatori sono raggruppati per ottimizzare l'esperienza del giocatore: puoi gestire il raggruppamento dei giocatori per adattarlo al meglio al tuo gioco utilizzando una serie di funzionalità. Consenti ai giocatori di cercare o sfogliare le sessioni di gioco o le richieste dei giocatori di gruppo quando arrivano in base al livello di abilità, alla latenza e ad altri attributi. Puoi gestire il raggruppamento dei giocatori da solo oppure puoi implementare il FlexMatch matchmaking con regole di partita personalizzabili.

- I giocatori possono partecipare alle partite attive: crea gruppi di giocatori per iniziare nuove sessioni di gioco o usa le sessioni dei giocatori per tenere traccia degli slot disponibili nelle sessioni esistenti e aggiungere nuovi giocatori. ConFlexMatch, riempi automaticamente le sessioni con giocatori ben assortiti.
- Gli amici possono giocare insieme come una squadra: il Player Party Support consente a gruppi di amici di giocare insieme. Le sessioni di gioco possono occupare i posti rimanenti con altri giocatori o limitare l'accesso. ConFlexMatch, gli amici possono formare squadre.
- I giocatori possono aggiungere dati personalizzati alle sessioni di gioco: includi informazioni sul gioco e sui giocatori nelle richieste di sessioni di gioco e nelle sessioni dei giocatori. Le informazioni vengono trasmesse al server di gioco per essere utilizzate in una sessione di gioco.
- I giocatori non aspettano all'infinito che arrivino le partite perfette: con FlexMatch il passare del tempo puoi ridurre i requisiti per le partite per evitare la frustrazione dei giocatori dovuta all'eccessiva coda. Crea regole di rilassamento progressivo che diano priorità agli aspetti più importanti della qualità delle partite.
- I giocatori possono scegliere di accettare o rifiutare le partite: conFlexMatch, puoi personalizzare i flussi di lavoro per consentire ai giocatori di accettare o rifiutare potenziali partite.

Offri connessioni affidabili durante tutto il gioco

I giocatori si aspettano un gameplay costante senza interruzioni o disconnessioni. Amazon GameLift Servers offre una gestione e una protezione complete delle sessioni:

- I giocatori godono di prestazioni di alta qualità sul AWS cloud: con l'hosting gestito da Amazon GameLift Servers, i giocatori ottengono server di gioco veloci e affidabili che funzionano su un'infrastruttura AWS informatica.
- Convalida i giocatori quando si connettono: abilitando le sessioni di gioco, puoi avere slot Amazon GameLift Servers riservati nelle sessioni di gioco e convalidare i giocatori al momento della connessione.
- I giocatori possono riconnettersi e mantenere i progressi di gioco: Amazon GameLift Servers supporta la riconnessione sia per le interruzioni lato server che lato client.
- Le partite dei giocatori sono protette dall'interruzione: la funzione opzionale di protezione della sessione di gioco impedisce l'interruzione delle sessioni attive durante eventi di scaling o altre interruzioni.

Migliora l'esperienza dei giocatori sulla base di dati del mondo reale

I giocatori si aspettano che i giochi funzionino in modo coerente e migliorino nel tempo. Amazon GameLift Servers fornisce un monitoraggio e un'analisi completi che aiutano a ottimizzare l'esperienza del giocatore:

- I giocatori subiscono meno interruzioni: il monitoraggio in tempo reale delle prestazioni della flotta, delle sessioni di gioco e dell'attività dei giocatori può identificare i problemi prima che abbiano un impatto significativo sul gameplay. Personalizza chiusure e migrazioni delle sessioni di gioco senza problemi.
- I giocatori traggono vantaggio dal monitoraggio proattivo dello stato dei server: Amazon GameLift Servers fornisce un monitoraggio continuo dello stato dei server e sostituisce automaticamente i server di gioco non integri per ridurre al minimo l'impatto dei guasti hardware o software. Configura i parametri del controllo dello stato di salute per supportare diversi requisiti di gioco.
- I giocatori ottengono miglioramenti del gameplay basati sui dati: sfrutta i registri delle sessioni di gioco e le analisi dettagliate per rivelare i modelli di comportamento dei giocatori e le prestazioni del server. Aggiungi il supporto di registrazione personalizzato per eventi specifici del gioco.

Integra funzionalità di gioco avanzate con altri servizi AWS

I giocatori desiderano funzionalità integrate come chat vocale, autenticazione sicura e progressi persistenti. Amazon GameLift Servers può essere integrato con altri AWS servizi per offrire esperienze di gioco complete:

- I giocatori ottengono un'autenticazione semplificata: l'integrazione con Amazon Cognito supporta diversi provider di identità e metodi di autenticazione, mantenendo al contempo le identità sicure dei giocatori durante le sessioni di gioco, con processi di accesso semplificati per i giocatori che ritornano.
- I giocatori possono mantenere i propri progressi di gioco tra le sessioni: l'integrazione con Amazon DynamoDB può essere utilizzata per archiviare i progressi dei giocatori, gli inventari e i dati persistenti con un accesso ad alte prestazioni che garantisce un impatto minimo sul gameplay e supporta la continuità tra le sessioni.
- I giocatori traggono vantaggio dalle informazioni analitiche: l'integrazione tra Amazon Kinesis e Amazon Simple Storage Service (Amazon S3) elabora l'analisi dei giochi per raccogliere informazioni sul comportamento e sulle preferenze dei giocatori, abilitando analisi in tempo reale.

che aiutano gli sviluppatori ad adattarsi ai cambiamenti dei modelli di gioco e a mantenere i giochi freschi e coinvolgenti.

Gestione dell'hosting di giochi in produzione con Amazon GameLift Servers

Dopo aver lanciato il gioco con Amazon GameLift Servers, dovrai gestire l'infrastruttura di hosting dei giochi per garantire prestazioni, affidabilità ed esperienza di gioco ottimali. Questo argomento tratta le funzionalità e gli strumenti principali per la gestione dei cicli di vita dell'hosting dei giochi durante la produzione.

Monitoraggio dello stato e delle prestazioni dell'hosting

Amazon GameLift Servers fornisce strumenti di monitoraggio completi per aiutarti a monitorare lo stato e le prestazioni della tua infrastruttura di hosting di giochi:

- **Metriche delle prestazioni:** monitora gli indicatori di prestazioni chiave per le flotte gestite, come l'utilizzo delle istanze, le sessioni dei giocatori e il posizionamento delle sessioni di gioco. Tieni traccia di queste metriche in Amazon GameLift Servers o in Amazon CloudWatch, dove puoi anche impostare allarmi per ricevere notifiche quando le metriche superano le soglie.
- **Metriche ed eventi della flotta:** monitora le metriche specifiche del parco veicoli, tra cui i processi attivi del server, le sessioni di gioco disponibili e le sessioni attive dei giocatori per garantire una capacità ottimale.
- **Metriche ed eventi relativi al posizionamento delle sessioni di gioco:** tieni traccia delle metriche e degli eventi per monitorare lo stato e le prestazioni del tuo sistema di posizionamento delle sessioni di gioco.
- **Registri delle sessioni di gioco:** accedi e analizza i registri delle sessioni di gioco per identificare problemi e comprendere i modelli di comportamento dei giocatori.
- **Realtimeregistri degli script:** per i giochi che utilizzano server in tempo reale, monitora l'esecuzione e le prestazioni degli script tramite registri dettagliati.
- **AWS Health Dashboard:** tieniti informato sullo stato AWS del servizio che potrebbe influire sulle tue Amazon GameLift Servers implementazioni.

Puoi accedere a questi strumenti di monitoraggio tramite la Amazon GameLift Servers console, la AWS CLI o utilizzare l' AWS SDK per Amazon GameLift Servers creare dashboard e soluzioni di monitoraggio personalizzate.

Gestione degli aggiornamenti e delle patch dei server di gioco

Mantenere aggiornati i server di gioco è fondamentale per la sicurezza, le prestazioni e l'aggiunta di nuove funzionalità. Amazon GameLift Servers offre diversi approcci per la gestione degli aggiornamenti:

- **Gestione delle build:** carica e gestisci più versioni delle build del tuo server di gioco. Ogni build ha una versione e può essere distribuita su flotte diverse.
- **Sostituzione della flotta:** crea nuove flotte con versioni AMI e aggiornamenti delle build dei server di gioco e sposta gradualmente il traffico dalle vecchie flotte a quelle nuove utilizzando code e alias di posizionamento delle sessioni di gioco.
- **Aggiornamenti degli script:** per i server in tempo reale, aggiorna gli script dei server senza sostituire l'intera flotta caricando nuove versioni degli script.
- **Aggiornamenti dei container:** per le distribuzioni basate su container, aggiorna le immagini e le definizioni dei container. Distribuisci nuove versioni dei tuoi server di gioco nelle flotte esistenti. Sostituisci le flotte per aggiornare le versioni AMI.
- **Implementazioni automatizzate:** utilizza AWS CodePipeline e crea CI/CD pipeline AWS CodeDeploy per gli aggiornamenti automatici dei server di gioco.

Quando pianifichi gli aggiornamenti, prendi in considerazione l'utilizzo di strategie di blue/green distribuzione per ridurre al minimo le interruzioni per i giocatori attivi e consentire un ripristino rapido se vengono rilevati problemi.

Ottimizzazione delle prestazioni e della scalabilità

Man mano che la tua base di giocatori si evolve, dovrai modificare la configurazione dell'hosting per mantenere prestazioni ottimali ed efficienza in termini di costi:

- **Scalabilità automatica:** configura le politiche di ridimensionamento della flotta in base a metriche come il numero di giocatori o l'utilizzo delle sessioni di gioco per regolare automaticamente la capacità.
- **Modifiche alla distribuzione regionale:** aggiungi o rimuovi aree dalle distribuzioni multiregionali in base ai dati demografici dei giocatori e ai requisiti di latenza.

- **Gestione delle code:** ottimizza le configurazioni della coda di posizionamento delle sessioni di gioco per bilanciare l'esperienza dei giocatori e i costi di hosting.
- **Selezione del tipo di istanza:** analizza le metriche delle prestazioni per determinare i tipi di istanza più convenienti per i requisiti del server di gioco.
- **Utilizzo delle istanze Spot:** sfrutta le istanze Spot per carichi di lavoro non critici per ridurre i costi, con strategie di fallback appropriate per le istanze On-Demand.
- **FlexMatchottimizzazione:** perfeziona le regole e gli algoritmi di matchmaking in base ai dati e al feedback effettivi dei giocatori.

Esamina regolarmente le CloudWatch metriche e i report sui costi per identificare opportunità di ottimizzazione e implementare le modifiche tramite la console o l'API. Amazon GameLift Servers

Risoluzione dei problemi e operazioni in tempo reale

Un'efficace risoluzione dei problemi e una gestione delle operazioni in tempo reale sono essenziali per mantenere un'esperienza positiva per i giocatori:

- **Notifiche degli eventi della flotta:** configura le notifiche di Amazon Simple Notification Service per eventi del parco veicoli come attività di ridimensionamento, cessazioni di istanze o errori nel posizionamento delle sessioni di gioco.
- **Debug del posizionamento delle sessioni di gioco:** utilizza registri di posizionamento dettagliati per identificare e risolvere problemi relativi al posizionamento delle sessioni di gioco.
- **Controlli dello stato dei processi del server:** monitora lo stato dei processi del server e sostituisci automaticamente i processi che non superano i controlli di integrità.
- **Accesso remoto:** connettiti alle istanze del parco istanze per la risoluzione diretta dei problemi. AWS Systems Manager Session Manager
- **Gestione degli alias:** utilizza gli alias per reindirizzare rapidamente il traffico dei giocatori lontano dalle flotte problematiche senza modificare le configurazioni dei client.
- **Backup e ripristino:** implementa backup regolari dei dati e della configurazione di gioco critici per consentire un rapido ripristino in caso di errori.

Stabilisci procedure di risposta agli incidenti e manuali chiari per i problemi più comuni per ridurre al minimo i tempi di inattività e l'impatto sui giocatori.

Strumenti e integrazione

Amazon GameLift Servers integra con vari AWS servizi e strumenti di terze parti per migliorare le capacità di gestione della produzione:

- **AWS CloudFormation**- Definisci e gestisci GameLift le tue risorse come infrastruttura come codice per implementazioni coerenti.
- **AWS Lambda**- Crea funzioni serverless per automatizzare le attività di gestione di routine e rispondere agli eventi.
- **Amazon EventBridge**: crea architetture basate sugli eventi che rispondono automaticamente ai cambiamenti del tuo ambiente. Amazon GameLift Servers
- **AWS Integrazione con SDK**: utilizzala AWS SDKs per creare strumenti di gestione e dashboard personalizzati in base alle tue esigenze specifiche.
- **Monitoraggio di terze parti**: integra piattaforme di monitoraggio e analisi di terze parti utilizzando l'esportazione delle CloudWatch metriche.

Sfrutta queste integrazioni per creare una soluzione di gestione completa in linea con i flussi di lavoro e i requisiti del tuo team.

Risorse correlate

- [Monitoraggio di Amazon GameLift con Amazon CloudWatch](#)
- [Aggiornamento delle GameLift flotte Amazon](#)
- [Scalare la capacità di hosting di giochi con Amazon GameLift](#)
- [Riferimento all'SDK Amazon GameLift Server](#)

Amazon GameLift Servertermini chiave

In questo argomento vengono fornite le definizioni dei termini tecnici chiave utilizzati nella Amazon GameLift Servers documentazione e nei servizi.

Sessione di gioco attiva

Una sessione di gioco attualmente in corso che accetta o ospita giocatori. Utilizzata come metrica per monitorare l'utilizzo della flotta e la pianificazione della capacità. Per informazioni, consulta [Monitora Amazon GameLift Servers con Amazon CloudWatch](#).

Alias

Un identificatore intuitivo per una flotta che fornisce un livello di astrazione tra il client di gioco e le risorse della flotta. Gli alias ti consentono di aggiornare le configurazioni della flotta senza modificare il codice del client di gioco. Per informazioni, consulta [Designazione astratta e Amazon GameLift Servers della flotta con un alias](#).

AMI (Amazon Machine Image)

Un'immagine di macchina virtuale preconfigurata utilizzata per creare EC2 istanze Amazon per flotte gestite. Contiene il sistema operativo, le librerie di runtime e il software degli Amazon GameLift Servers agenti necessari per eseguire i server di gioco. Per informazioni, consulta [Amazon GameLift Servers EC2 flotte gestite](#).

Flotta ovunque

Un tipo di flotta che utilizza risorse di elaborazione fornite dal cliente gestite all'esterno di Amazon GameLift Servers. Le flotte Anywhere consentono di integrare l'infrastruttura esistente o le soluzioni cloud ibride con Amazon GameLift Servers la gestione delle sessioni di gioco. Per informazioni, consulta [Amazon GameLift Servers Flotte ovunque](#).

Authentication token (Token di autenticazione)

Una credenziale di sicurezza generata da Amazon GameLift Servers che autentica i processi del server di gioco in esecuzione su risorse di calcolo Anywhere. Necessaria per consentire ai processi del server di comunicare con il servizio. Amazon GameLift Servers Per informazioni, consulta [Amazon GameLift Servers Flotte ovunque](#).

Servizio di backend

Un livello di coordinamento tra i client di gioco e il Amazon GameLift Servers servizio che controlla la comunicazione per le richieste di posizionamento delle sessioni di gioco e recupera le informazioni sulla sessione di gioco. Funge da intermediario sicuro per proteggere le risorse di hosting da attacchi dannosi. Per informazioni, consulta [Come Amazon GameLift Servers funziona l'hosting con](#).

Costruzione del server di gioco

Una versione in pacchetto del software del server di gioco che include tutti i file necessari per far funzionare il server di gioco su risorse Amazon GameLift Servers di hosting. Contiene il file eseguibile, le dipendenze e le risorse del server di gioco. Per informazioni, consulta [Package di un server di gioco costruito per la distribuzione](#).

Calcolo

Nelle flotte di container, rappresenta una copia di un gruppo di contenitori di server di gioco su un'istanza della flotta. Ogni computer può ospitare una sessione di gioco alla volta. Per informazioni, consulta [Come funzionano i contenitori Amazon GameLift Servers](#).

Flotta di container

Un tipo di flotta gestita che gestisce server di gioco containerizzati utilizzando immagini di container anziché build tradizionali. Fornisce flessibilità per l'implementazione di server di gioco con ambienti di runtime e dipendenze specifici. Per informazioni, consulta [Amazon GameLift Servers](#) [flotte di container gestite](#).

Gruppo di contenitori

Una raccolta di contenitori che interagiscono per fornire funzionalità di server di gioco. Simile a un'attività o un contenitore, che definisce il comportamento dei container, le dipendenze e la condivisione delle risorse all'interno di una flotta di container. Per informazioni, consulta [Come funzionano i contenitori Amazon GameLift Servers](#).

Destinazione

Nelle code delle sessioni di gioco, una flotta o un alias in grado di ricevere richieste di posizionamento per le sessioni di gioco. Le code possono avere più destinazioni con priorità e politiche di latenza diverse. Per informazioni, consulta [Personalizza una coda di sessioni di gioco](#).

Parco istanze

Una raccolta di risorse informatiche che gestiscono i tuoi server di gioco. Le risorse della flotta possono essere distribuite su più aree geografiche e ogni risorsa esegue uno o più processi del server di gioco. Per informazioni, consulta [Implementa flotte di hosting per Amazon GameLift Servers](#).

Ubicazione della flotta

Una AWS regione o una zona locale specifica in cui vengono distribuite le istanze della flotta. Le flotte con più sedi possono estendersi su più sedi per ridurre la latenza e migliorare la disponibilità. Per informazioni, consulta [Implementa flotte di hosting per Amazon GameLift Servers](#).

FlexMatch

Amazon GameLift Servers servizio di matchmaking che forma automaticamente le partite dei giocatori in base a regole configurabili e richiede il posizionamento delle sessioni di gioco per le

partite completate. Supporta l'abbinamento basato sulle abilità, il bilanciamento della squadra e il backfill delle partite. [Vedi Cos'è? GameLift FlexMatch](#)

Client di gioco

Il tuo software di gioco che funziona sul dispositivo di un giocatore. Avvia il posizionamento dei giocatori nelle sessioni di gioco comunicando con un servizio di backend e si connette direttamente ai server di gioco per il gioco. Per informazioni, consulta [Come Amazon GameLift Servers funziona l'hosting con](#).

Proprietà del gioco

Coppie chiave-valore di dati personalizzati associati a una sessione di gioco. Utilizzato per trasmettere informazioni di configurazione, regole di gioco o altri metadati ai server e ai client di gioco. Vedi [GameProperty](#) nel riferimento all'API.

Server di gioco

Il tuo software server personalizzato che gestisce lo stato del gioco, elabora le azioni dei giocatori e sincronizza il gameplay tra più giocatori connessi. Si integra con Amazon GameLift Servers Server SDK per comunicare con il servizio. Per informazioni, consulta [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#).

Attivazione del server di gioco

Processo di avvio e inizializzazione di un processo del server di gioco su una risorsa di hosting. Include il caricamento del file eseguibile del server di gioco, l'instaurazione della comunicazione con Amazon GameLift Servers e la preparazione per ospitare sessioni di gioco. Per informazioni, consulta [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#).

Contenitore del server di gioco

Un contenitore all'interno di un gruppo di contenitori di server di gioco che esegue l'applicazione del server di gioco. Designato automaticamente come essenziale per il ciclo di vita del gruppo di contenitori. Per informazioni, consulta [Come funzionano i contenitori Amazon GameLift Servers](#).

Gruppo di contenitori di server di gioco

Un tipo di gruppo di contenitori che gestisce i contenitori che eseguono applicazioni per server di gioco e software di supporto. È necessario per consentire alle flotte di container di ospitare sessioni di gioco e può essere replicato tra le istanze del parco veicoli. Per informazioni, consulta [Come funzionano i contenitori Amazon GameLift Servers](#).

Processo del server di gioco

Un'istanza in esecuzione del server di gioco eseguibile su una risorsa di hosting. Ogni processo può ospitare una sessione di gioco alla volta e comunica direttamente con lui Amazon GameLift Servers per segnalare lo stato e ricevere istruzioni. Per informazioni, consulta [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#).

Sessione di gioco

Un'istanza del gioco multiplayer in esecuzione su un processo del server di gioco. Rappresenta un'unica esperienza di gioco a cui più giocatori possono unirsi e partecipare insieme. Vedi [Come Amazon GameLift Servers funziona l'hosting con](#) e [GameSession](#) nell'API Reference.

Dati della sessione di gioco

Informazioni associate a una sessione di gioco, tra cui dati del giocatore, stato del gioco, proprietà della sessione e altre informazioni di runtime utilizzate dai server e dai client di gioco. Vedi [GameSession](#) nell'API Reference.

Posizionamento della sessione di gioco

Il processo di localizzazione dei server di gioco disponibili per ospitare nuove sessioni di gioco. Utilizza informazioni in tempo reale sulla disponibilità dei server di gioco per prendere decisioni di posizionamento ottimali in base a fattori quali costi e latenza. Vedi [Integra la funzionalità del client di Amazon GameLift Servers gioco](#) e [StartGameSessionPlacement](#) nell'API Reference.

Protezione della sessione di gioco

Un'impostazione che Amazon GameLift Servers impedisce di terminare i processi del server di gioco che ospitano sessioni di gioco attive durante gli eventi di scalabilità o gli aggiornamenti della flotta. Garantisce che i giochi in corso non vengano interrotti. Per informazioni, consulta [Imposta manualmente la capacità per una Amazon GameLift Servers flotta](#).

Coda delle sessioni di gioco

Un meccanismo di posizionamento che utilizza algoritmi configurabili per organizzare sessioni di gioco e ottenere risultati ottimali. Può dare priorità ai posizionamenti in base a fattori quali il costo di hosting più basso e la latenza più bassa dei giocatori in più aree geografiche. Per informazioni, consulta [Configura il posizionamento delle sessioni di gioco](#).

Istanza inattiva

Un'istanza fleet che è in esecuzione ma che attualmente non ospita sessioni di gioco attive. Utilizzata come metrica per il monitoraggio dell'efficienza della flotta e la scalabilità delle decisioni. Per informazioni, consulta [Monitora Amazon GameLift Servers con Amazon CloudWatch](#).

Autorizzazioni in entrata

Regole di accesso alla rete che definiscono quali indirizzi IP e intervalli di porte possono essere utilizzati per connettersi alle istanze del parco istanze. Controlla l'accesso ai server di gioco e garantisce connessioni sicure. Sia le flotte di container gestite che quelle gestite richiedono la configurazione delle autorizzazioni in entrata. EC2

Consulta [IpPermission](#) e [Configurare le connessioni di rete](#).

Politica di latenza

Regole nelle code delle sessioni di gioco che definiscono soglie di latenza accettabili per le connessioni dei giocatori. Utilizzato per garantire che i giocatori partecipino a sessioni di gioco con prestazioni di rete accettabili. Per informazioni, consulta [Personalizza una coda di sessioni di gioco](#).

Flotta gestita

Un tipo di flotta in cui Amazon GameLift Servers distribuisce e gestisce risorse informatiche in. Cloud AWS Fornisce scalabilità automatica, monitoraggio dello stato e gestione dell'infrastruttura. Per informazioni, consulta [Amazon GameLift Servers EC2 flotte gestite](#).

Matchmaking

Il processo di raggruppamento dei giocatori per il gioco multiplayer in base a criteri come il livello di abilità, latenza o altri fattori specifici del gioco. Nel, gestito principalmente Amazon GameLift Servers da. FlexMatch Vedi [Cos'è GameLift FlexMatch?](#)

Gruppo di metriche

Una raccolta di metriche prestazionali correlate utilizzate per il monitoraggio e l'auto-scaling delle decisioni. Raggruppa metriche come sessioni di gioco attive, server di gioco disponibili e utilizzo dei giocatori. Per informazioni, consulta [Capacità di dimensionamento automatico del parco istanze con Amazon GameLift Servers](#).

Gruppo di contenitori per istanza

Un tipo di gruppo di contenitori opzionale che esegue software aggiuntivo su ogni istanza del parco veicoli. Utile per servizi in background o strumenti di monitoraggio, con una sola copia distribuita per istanza. Per informazioni, consulta [Come funzionano i contenitori Amazon GameLift Servers](#).

Sessione del giocatore

Uno slot di connessione per un giocatore in una sessione di gioco. Utilizzato per prenotare gli slot delle sessioni di gioco, convalidare i giocatori quando si connettono e tenere traccia dello stato di connessione e delle disconnessioni dei giocatori. Vedi [Integra la funzionalità del client di Amazon GameLift Servers gioco](#) e [PlayerSession](#) nell'API Reference.

Configurazione prioritaria

Impostazioni che personalizzano l'ordine di priorità di una coda di sessioni di gioco quando si selezionano le destinazioni per il posizionamento della sessione di gioco. Definisce il comportamento generale della coda, a differenza degli elenchi di priorità prioritaria che modificano la priorità per le singole richieste di posizionamento. Per informazioni, consulta [Dare la priorità al posizionamento delle sessioni di gioco](#).

Elenco di priorità prioritarie

Un elenco alternativo di luoghi prioritari che possono essere inclusi in una richiesta di posizionamento per una sessione di gioco. Sostituisce in modo efficace la prioritizzazione configurata della coda per le posizioni solo per quella richiesta, senza influire sulle altre richieste. Per informazioni, consulta [Dare la priorità al posizionamento delle sessioni di gioco](#).

Gestore dei processi

Un componente che gestisce il ciclo di vita di più processi del server di gioco all'interno di un contenitore o di un'istanza. Gestisce l'avvio, l'arresto e il monitoraggio dei singoli processi del server di gioco. Per informazioni, consulta [Ottimizza la configurazione del runtime del server di gioco su server gestiti Amazon GameLift Servers](#).

Script in tempo reale

JavaScript codice che definisce la logica di gioco per i server in Amazon GameLift Servers tempo reale. Gestisce le connessioni dei giocatori, il routing dei messaggi e la gestione dello stato del gioco senza richiedere un server di gioco personalizzato. Scopri [Cos'è Amazon GameLift Realtime Servers?](#)

Strategia di routing

Configurazione che determina in che modo gli alias indirizzano il traffico verso le flotte. Può indirizzare verso una flotta specifica o distribuire il traffico su più flotte in base a regole definite. Per informazioni, consulta [Designazione astratta e Amazon GameLift Servers della flotta con un alias](#).

Configurazione di runtime

Impostazioni che definiscono il modo in cui i processi dei server di gioco vengono eseguiti sulle istanze della flotta, tra cui il numero di processi simultanei, i parametri di avvio e l'allocazione delle risorse. Per informazioni, consulta [Ottimizza la configurazione del runtime del server di gioco su server gestiti Amazon GameLift Servers](#).

Dimensionamento

Il processo di regolazione della capacità della flotta mediante l'aggiunta o la rimozione di risorse di hosting in base alla domanda dei giocatori. Può essere automatico (in base alle metriche di utilizzo) o manuale (controllato dall'amministratore). Per informazioni, consulta [Capacità di dimensionamento automatico del parco istanze con Amazon GameLift Servers](#).

Politica di scalabilità

Regole che definiscono quando e come regolare automaticamente la capacità della flotta. Può essere basato su obiettivi (mantenimento di un utilizzo specifico) o basato su regole (scala basata su soglie metriche). Per informazioni, consulta [Capacità di dimensionamento automatico del parco istanze con Amazon GameLift Servers](#).

SDK per server

Kit di sviluppo software che fornisce gli strumenti e le librerie necessari per integrare il tuo server di gioco con Amazon GameLift Servers. Consente la comunicazione tra il server di gioco e il servizio per la gestione delle sessioni, i report sullo stato di salute e gli aggiornamenti di stato. Per informazioni, consulta [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#).

Multigiocatore basato su sessioni

Un'architettura di gioco multiplayer in cui il gioco si svolge all'interno di sessioni discrete e limitate nel tempo. Ogni sessione ha un inizio e una fine definiti, con i giocatori che si uniscono e se ne vanno secondo necessità. Per informazioni, consulta [Come Amazon GameLift Servers funziona l'hosting con](#).

Istanze Spot

Un tipo di EC2 istanza Amazon che utilizza AWS capacità inutilizzata a costi ridotti. Le istanze Spot possono essere interrotte quando viene ripristinata la capacità AWS necessaria, il che le rende adatte a carichi di lavoro di gioco con tolleranza ai guasti. Per informazioni, consulta [Crea una coda per le istanze Spot](#).

Certificato TLS

Un certificato digitale utilizzato per consentire la comunicazione crittografata tra client e server di gioco. Necessario per le flotte abilitate al TLS per garantire una trasmissione sicura dei dati. Per informazioni, consulta [Crea una EC2 flotta Amazon GameLift Servers gestita](#).

Nozioni di base su Amazon GameLift Servers

Sfrutta queste risorse introduttive per saperne di più sul Amazon GameLift Servers servizio e su come iniziare a sviluppare una soluzione di hosting personalizzata per i tuoi giochi multiplayer basati su sessioni.

Prima di iniziare

- Creare un Account AWS (o designare uno esistente) con cui utilizzarlo. Amazon GameLift Servers
- Configurare gli utenti con le autorizzazioni Amazon GameLift Servers e i servizi correlati AWS .
- Selezionare un utente Regione AWS su cui lavorare. Per lo sviluppo, scegli una regione vicina alla tua posizione. Puoi cambiare regione in qualsiasi momento.

[Configurare un account AWS utente](#)

Scegli il tuo percorso

Voglio esplorare Amazon GameLift Servers velocemente

Ideale per: apprendimento, creazione di un proof of concept, prototipazione rapida

Opzioni di avvio rapido:

- [Wrapper per server di gioco per Amazon GameLift Servers: vedi. Tutorial: onboarding rapido con il wrapper Amazon GameLift Servers](#) Questo strumento è il modo più semplice e veloce per ospitare ed eseguire sessioni di gioco sul server di gioco Amazon GameLift Servers, senza dover modificare il codice di gioco. Il game server wrapper offre funzionalità di base per la gestione delle sessioni di gioco e un'implementazione semplificata del server di gioco. È ideale per effettuare una valutazione pratica dell'Amazon GameLift Servers utilizzo del proprio progetto di gioco o di un progetto di esempio. Quando sei pronto per creare una soluzione di hosting di giochi personalizzata, passa a una delle opzioni di sviluppo personalizzate con piena integrazione con l'SDK del server per Amazon GameLift Servers. Se il tuo gioco non necessita di una soluzione di hosting personalizzata, puoi continuare a utilizzare il wrapper del server di gioco per distribuire e ospitare i server di gioco in produzione.

- [Amazon GameLift Servers plug-in per Unreal Engine o Unity](#): i plugin offrono flussi di lavoro con interfaccia grafica e risorse di esempio per guidarti nei passaggi iniziali e implementare il tuo server di gioco con una soluzione di hosting di base. Utilizza il plug-in per configurare l'hosting con flotte Anywhere autogestite o implementare flotte EC2 o flotte di container gestite e basate sul cloud. Quando sei pronto a sviluppare una soluzione di hosting personalizzata, puoi sfruttare le soluzioni integrate nei plug-in.
- [Starter kit per container Amazon GameLift Servers gestiti](#): questo kit semplifica le attività per integrare un server di gioco, preparare l'immagine di un container del server di gioco e implementare una flotta di container per l'hosting. Per quanto riguarda l'integrazione, il kit aggiunge funzionalità essenziali per la gestione delle sessioni di gioco al server di gioco. Il kit utilizza modelli preconfigurati per creare una flotta di container e una pipeline di distribuzione automatizzata per il server di gioco. Quando sei pronto per aggiungere funzionalità complete di gestione delle sessioni di gioco, segui una delle roadmap di sviluppo personalizzate per integrare l'SDK del server. Amazon GameLift Servers

Voglio creare una soluzione di hosting personalizzata

Ideale per: creare una soluzione a livello di produzione per un server di gioco personalizzato

Controlla la pagina [Amazon GameLift Servers opzioni di hosting di giochi](#) e scegli una delle opzioni per la tua soluzione. Segui una di queste roadmap di sviluppo per iniziare a creare una soluzione di hosting personalizzata completa per il tuo gioco. Le tabelle di marcia forniscono indicazioni dettagliate su come creare, testare e personalizzare ogni componente della tua soluzione di hosting.

- [Roadmap di sviluppo per l'hosting con EC2 Amazon GameLift Servers gestito](#)
- [Roadmap di sviluppo per l'hosting con contenitori Amazon GameLift Servers gestiti](#)
- [Roadmap di sviluppo per l'hosting con Anywhere Amazon GameLift Servers](#)
- [Roadmap di sviluppo per l'hosting ibrido con Amazon GameLift Servers](#)

Percorso di apprendimento consigliato

1. Inizia in piccolo. Prova il wrapper del server di gioco o il plugin per Unreal.
2. Comprendi i concetti. Rivedi la pagina [Come Amazon GameLift Servers funziona l'hosting con](#).
3. Scegli l'architettura. Seleziona il tuo modello di hosting in base ai requisiti del gioco.

4. Costruisci e testa. Segui la roadmap di sviluppo appropriata. Crea una versione base di ogni componente, quindi esegui iterazioni e personalizza.
5. Ridimensiona e ottimizza. Ottimizza la tua soluzione per l'utilizzo a livello di produzione. Aggiungi funzionalità avanzate come il matchmaking.

Configurare un account AWS utente

Tip

Usa questi argomenti per ottenere assistenza con queste attività:

- Acquistane uno nuovo Account AWS da utilizzare con Amazon GameLift Servers.
- Crea un utente o un gruppo con le autorizzazioni per lavorare con Amazon GameLift Servers le risorse.
- Imposta le credenziali di sicurezza (ti servono per usare AWS CLI gli strumenti e i Amazon GameLift Servers plugin per Unreal e Unity)
-

Come per tutti i AWS servizi, è necessario un utente Account AWS per utilizzare il Amazon GameLift Servers servizio e gli strumenti. An Account AWS svolge due funzioni principali: (1) fornisce un contenitore per tutte le AWS risorse create con l'account; e (2) consente di gestire la sicurezza AWS delle risorse, inclusa la configurazione dell'autenticazione degli utenti e il controllo delle autorizzazioni di accesso degli utenti. Non è previsto alcun costo per la creazione di un Account AWS.

Esplora Amazon GameLift Servers con o senza un AWS account

Non hai bisogno di un AWS account per:

- Scopri AWS gli strumenti per creare, gestire e far crescere le esperienze di gioco su [AWS for Games](#). Leggi il [blog](#) e sfoglia la [libreria Solutions for Games](#).
- Scopri di più Amazon GameLift Servers nella [panoramica del prodotto FAQs e nelle risorse](#). Chiedi AWS di trovare le risposte alle tue domande sui prodotti. (Prova questo: «Sto cercando opzioni a basso costo per ospitare la mia partita multiplayer».)
- Per un'analisi più approfondita, scopri cosa Amazon GameLift Servers funziona nella [documentazione tecnica](#), comprese le guide per gli sviluppatori per l'hosting e il matchmaking e la guida di riferimento sulle API di servizio.

- Consulta le informazioni sulle tecniche di ottimizzazione [Amazon GameLift Servers dei prezzi](#) e dei costi. Prova il [Calcolatore dei prezzi per scoprire come vengono calcolati](#) i costi di hosting in base al picco di utilizzo simultaneo dei giocatori (CCU).
- Scarica i download e visualizza gli archivi di codice Amazon GameLift Servers SDKs, i plugin e i toolkit. [Vedi Guida introduttiva. Amazon GameLift Servers](#) (Hai bisogno di un Account AWS per usarli.)

È necessario un AWS account per:

- Segui i flussi di lavoro di onboarding con i Amazon GameLift Servers plugin per Unreal e Unity o usa il wrapper del server di gioco.
- Crea e gestisci risorse utilizzando. AWS Console di gestione AWS
- Crea e gestisci AWS risorse utilizzando AWS Command Line Interface.
- Usa Amazon Q con la documentazione Amazon GameLift Servers tecnica inclusa per trovare risposte, linee guida e consigli.

Argomenti

- [Registrati per un Account AWS](#)
- [Crea un utente con accesso amministrativo](#)
- [Imposta le autorizzazioni utente per Amazon GameLift Servers](#)
- [Configura l'accesso programmatico per gli utenti](#)
- [Configura l'accesso programmatico per il tuo gioco](#)
- [Esempi di autorizzazioni IAM per Amazon GameLift Servers](#)
- [Configurare un ruolo di servizio IAM per Amazon GameLift Servers](#)

Registrati per un Account AWS

Se non ne hai uno Account AWS, completa i seguenti passaggi per crearne uno.

Per iscriverti a un Account AWS

1. Apri la <https://portal.aws.amazon.com/billing/registrazione>.
2. Segui le istruzioni online.

Nel corso della procedura di registrazione riceverai una telefonata o un messaggio di testo e ti verrà chiesto di inserire un codice di verifica attraverso la tastiera del telefono.

Quando ti iscrivi a un Account AWS, Utente root dell'account AWS viene creato un. L'utente root dispone dell'accesso a tutte le risorse e tutti i Servizi AWS nell'account. Come best practice di sicurezza, assegna l'accesso amministrativo a un utente e utilizza solo l'utente root per eseguire [attività che richiedono l'accesso di un utente root](#).

AWS ti invia un'email di conferma dopo il completamento della procedura di registrazione. In qualsiasi momento, puoi visualizzare l'attività corrente del tuo account e gestirlo accedendo a <https://aws.amazon.com/> e scegliendo Il mio account.

Crea un utente con accesso amministrativo

Dopo esserti registrato Account AWS, proteggi Utente root dell'account AWS AWS IAM Identity Center, abilita e crea un utente amministrativo in modo da non utilizzare l'utente root per le attività quotidiane.

Proteggi i tuoi Utente root dell'account AWS

1. Accedi [Console di gestione AWS](#) come proprietario dell'account scegliendo Utente root e inserendo il tuo indirizzo Account AWS email. Nella pagina successiva, inserisci la password.

Per informazioni sull'accesso utilizzando un utente root, consulta la pagina [Accedere come utente root](#) nella Guida per l'utente di Accedi ad AWS .

2. Abilita l'autenticazione a più fattori (MFA) per l'utente root.

Per istruzioni, consulta [Abilitare un dispositivo MFA virtuale per l'utente Account AWS root \(console\)](#) nella Guida per l'utente IAM.

Crea un utente con accesso amministrativo

1. Abilita il Centro identità IAM.

Per istruzioni, consulta [Abilitazione del AWS IAM Identity Center](#) nella Guida per l'utente di AWS IAM Identity Center .

2. Nel Centro identità IAM, assegna l'accesso amministrativo a un utente.

Per un tutorial sull'utilizzo di IAM Identity Center directory come fonte di identità, consulta [Configurare l'accesso utente con l'impostazione predefinita IAM Identity Center directory](#) nella Guida per l'AWS IAM Identity Center utente.

Accesso come utente amministratore

- Per accedere come utente del Centro identità IAM, utilizza l'URL di accesso che è stato inviato al tuo indirizzo e-mail quando hai creato l'utente del Centro identità IAM.

Per informazioni sull'accesso utilizzando un utente IAM Identity Center, consulta [AWS Accedere al portale di accesso](#) nella Guida per l'Accedi ad AWS utente.

Assegnazione dell'accesso ad altri utenti

1. Nel Centro identità IAM, crea un set di autorizzazioni conforme alla best practice per l'applicazione di autorizzazioni con il privilegio minimo.

Segui le istruzioni riportate nella pagina [Creazione di un set di autorizzazioni](#) nella Guida per l'utente di AWS IAM Identity Center .

2. Assegna al gruppo prima gli utenti e poi l'accesso con autenticazione unica (Single Sign-On).

Per istruzioni, consulta [Aggiungere gruppi](#) nella Guida per l'utente di AWS IAM Identity Center .

Imposta le autorizzazioni utente per Amazon GameLift Servers

Crea utenti aggiuntivi o estendi le autorizzazioni di accesso agli utenti esistenti in base alle esigenze delle tue Amazon GameLift Servers risorse. Come [best practice \(best practice di sicurezza in IAM\)](#), applica le autorizzazioni con privilegi minimi per tutti gli utenti. Per indicazioni sulla sintassi delle autorizzazioni, consulta. [Esempi di autorizzazioni IAM per Amazon GameLift Servers](#)

Utilizza le seguenti istruzioni per impostare le autorizzazioni utente in base a come gestisci gli utenti del tuo account. AWS

Per fornire l'accesso, aggiungi autorizzazioni agli utenti, gruppi o ruoli:

- Utenti e gruppi in AWS IAM Identity Center:

Crea un set di autorizzazioni. Segui le istruzioni riportate nella pagina [Create a permission set](#) (Creazione di un set di autorizzazioni) nella Guida per l'utente di AWS IAM Identity Center .

- Utenti gestiti in IAM tramite un provider di identità:

Crea un ruolo per la federazione delle identità. Segui le istruzioni riportate nella pagina [Create a role for a third-party identity provider \(federation\)](#) della Guida per l'utente IAM.

- Utenti IAM:
 - Crea un ruolo che l'utente possa assumere. Segui le istruzioni riportate nella pagina [Create a role for an IAM user](#) della Guida per l'utente IAM.
 - (Non consigliato) Collega una policy direttamente a un utente o aggiungi un utente a un gruppo di utenti. Segui le istruzioni riportate nella pagina [Aggiunta di autorizzazioni a un utente \(console\)](#) nella Guida per l'utente IAM.

Quando lavori con utenti IAM, come best practice, assegna sempre le autorizzazioni ai ruoli o ai gruppi di utenti, non ai singoli utenti.

Configura l'accesso programmatico per gli utenti

Gli utenti necessitano dell'accesso programmatico se desiderano interagire con l' AWS esterno di. Console di gestione AWS Il modo per concedere l'accesso programmatico dipende dal tipo di utente che accede. AWS

Per fornire agli utenti l'accesso programmatico, scegli una delle seguenti opzioni.

Quale utente necessita dell'accesso programmatico?	Per	Come
IAM	(Consigliato) Utilizza le credenziali della console come credenziali temporanee per firmare le richieste programmatiche a,, o. AWS CLI AWS SDKs AWS APIs	Segui le istruzioni per l'interfaccia che desideri utilizzare. • Per la AWS CLI, consulta Login for AWS local development nella Guida per l'AWS Command Line Interface utente.

Quale utente necessita dell'accesso programmatico?	Per	Come
		• Per AWS SDKs, consulta Login for AWS local development nella AWS SDKs and Tools Reference Guide.
Identità della forza lavoro (Utenti gestiti nel centro identità IAM)	Utilizza credenziali temporane e per firmare le richieste programmatiche a AWS CLI, AWS SDKs, o. AWS APIs	Segui le istruzioni per l'interfaccia che desideri utilizzare. • Per la AWS CLI, vedere Configurazione dell'uso AWS IAM Identity Center nella AWS CLI Guida per l'utente. AWS Command Line Interface • Per AWS SDKs gli strumenti e AWS APIs, consulta l'autenticazione di IAM Identity Center nella Guida di riferimento AWS SDKs and Tools.
IAM	Utilizza credenziali temporane e per firmare le richieste programmatiche a AWS CLI, AWS SDKs, o. AWS APIs	Seguendo le istruzioni riportate in Utilizzo delle credenziali temporanee con le AWS risorse nella Guida per l'utente IAM.

Quale utente necessita dell'accesso programmatico?	Per	Come
IAM	(Non consigliato) Utilizza credenziali a lungo termine per firmare richieste programmatiche a AWS CLI,, AWS SDKs o. AWS APIs	Segui le istruzioni per l'interfaccia che desideri utilizzare. • Per la AWS CLI, consulta Autenticazione tramite credenziali utente IAM nella Guida per l'utente.AWS Command Line Interface • Per gli strumenti AWS SDKs e gli strumenti, consulta Autenticazione tramite credenziali a lungo termine nella Guida di riferimento agli strumenti e agli AWS SDKs strumenti. • Per AWS APIs, consulta la sezione Gestione delle chiavi di accesso per gli utenti IAM nella Guida per l'utente IAM.

Se utilizzi le chiavi di accesso, consulta [Best practice per la gestione delle chiavi di AWS accesso](#).

Configura l'accesso programmatico per il tuo gioco

La maggior parte dei giochi utilizza servizi di backend per comunicare Amazon GameLift Servers tramite. AWS SDKs Utilizza un servizio di backend (che funge da client di gioco) per richiedere sessioni di gioco, inserire giocatori in giochi e altre attività. Questi servizi richiedono l'accesso programmatico e le credenziali di sicurezza per autenticare le chiamate all'API del servizio. Amazon GameLift Servers

Infatti Amazon GameLift Servers, gestisci questo accesso creando un utente giocatore in AWS Identity and Access Management (IAM). Gestisci le autorizzazioni utente del giocatore tramite una delle seguenti opzioni:

- Crea un ruolo IAM con le autorizzazioni dell'utente giocatore e consenti all'utente giocatore di assumere il ruolo quando necessario. Il servizio di backend deve includere il codice per assumere questo ruolo prima di effettuare richieste a. Amazon GameLift Servers In conformità con le migliori pratiche di sicurezza, i ruoli forniscono un accesso limitato e temporaneo. Puoi utilizzare i ruoli per i carichi di lavoro in esecuzione su AWS risorse ([ruoli IAM](#)) o all'esterno di AWS ([IAM Roles Anywhere](#)).
- Crea un gruppo di utenti IAM con autorizzazioni utente giocatore e aggiungi il tuo utente giocatore al gruppo. Questa opzione fornisce all'utente giocatore le credenziali a lungo termine, che il servizio di backend deve archiviare e utilizzare per comunicare. Amazon GameLift Servers

Per la sintassi della politica di autorizzazione, vedi. [Esempi di autorizzazioni utente per i giocatori](#)

Per ulteriori informazioni sulla gestione delle autorizzazioni per l'uso da parte di un carico di lavoro, consulta [IAM Identities: Temporary credentials in IAM](#).

Esempi di autorizzazioni IAM per Amazon GameLift Servers

Usa la sintassi in questi esempi per impostare le autorizzazioni AWS Identity and Access Management (IAM) per gli utenti che devono accedere alle Amazon GameLift Servers risorse. Per ulteriori informazioni sulla gestione delle autorizzazioni degli utenti, vedere. [Imposta le autorizzazioni utente per Amazon GameLift Servers](#) Quando gestisci le autorizzazioni per utenti esterni all'IAM Identity Center, come best practice, assegna sempre le autorizzazioni ai ruoli o ai gruppi di utenti IAM, non ai singoli utenti.

Se la utilizzi Amazon GameLift Servers FleetIQ come soluzione autonoma, consulta [Configurare](#) il tuo modulo. Account AWS Amazon GameLift Servers FleetIQ

Esempi di autorizzazioni amministrative

Questi esempi forniscono a un amministratore di hosting o a uno sviluppatore un accesso mirato alla gestione delle risorse di hosting dei Amazon GameLift Servers giochi.

Example Sintassi per le autorizzazioni Amazon GameLift Servers complete alle risorse di accesso

L'esempio seguente estende l'accesso completo a tutte le Amazon GameLift Servers risorse.

JSON

```
{
```



```
"Version": "2012-10-17",
"Statement": {
  "Effect": "Allow",
  "Action": "gamelift:*",
  "Resource": "*"
}
```

Example Sintassi per le autorizzazioni Amazon GameLift Servers delle risorse con supporto per le regioni che non sono abilitate per impostazione predefinita

L'esempio seguente estende l'accesso a tutte le Amazon GameLift Servers risorse e le AWS regioni che non sono abilitate per impostazione predefinita. Per ulteriori informazioni sulle regioni che non sono abilitate per impostazione predefinita e su come attivarle, vedi [Gestione Regioni AWS](#) in Riferimenti generali di AWS.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "ec2:DescribeRegions",
      "gamelift:*"
    ],
    "Resource": "*"
  }
}
```

Example Sintassi della Amazon GameLift Servers risorsa per accedere alle immagini dei contenitori in Amazon ECR

L'esempio seguente estende l'accesso alle azioni Amazon GameLift Servers di Amazon Elastic Container Registry (Amazon ECR) necessarie agli utenti quando lavorano con flotte di container gestite.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "ecr:DescribeImages",
      "ecr:BatchGetImage",
      "ecr:GetDownloadUrlForLayer"
    ],
    "Resource": "*"
  }
}
```

Example Sintassi delle risorse e delle Amazon GameLift Servers autorizzazioni **PassRole**

L'esempio seguente estende l'accesso a tutte le Amazon GameLift Servers risorse e consente a un utente di passare un ruolo di servizio IAM a. Amazon GameLift Servers Un ruolo di servizio offre una capacità Amazon GameLift Servers limitata di accedere ad altre risorse e servizi per conto dell'utente, come descritto in [Configurare un ruolo di servizio IAM per Amazon GameLift Servers](#). Ad esempio, quando si risponde a una CreateBuild richiesta, è Amazon GameLift Servers necessario accedere ai file di build in un bucket Amazon S3. Per ulteriori informazioni sull'PassRoleazione, consulta [IAM: Pass an IAM role to a specific AWS service](#) nella IAM User Guide.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "gamelift:*",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:PassRole",
      "Resource": "*",
    }
  ]
}
```

```

    "Condition": {
      "StringEquals": {
        "iam:PassedToService": "gamelift.amazonaws.com"
      }
    }
  }
]
}

```

Esempi di autorizzazioni utente per i giocatori

Questi esempi consentono a un servizio di backend o a un'altra entità di effettuare chiamate API all'Amazon GameLift Servers API. Coprono gli scenari comuni per la gestione delle sessioni di gioco, delle sessioni dei giocatori e del matchmaking. Per ulteriori dettagli, consultare [Configura l'accesso programmatico per il tuo gioco](#).

Example Sintassi per i permessi di posizionamento delle sessioni di gioco

L'esempio seguente estende l'accesso alle code di posizionamento delle sessioni di gioco Amazon GameLift Servers APIs che utilizzano le code di posizionamento delle sessioni di gioco per creare sessioni di gioco e gestire le sessioni dei giocatori.

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PlayerPermissionsForGameSessionPlacements",
      "Effect": "Allow",
      "Action": [
        "gamelift:StartGameSessionPlacement",
        "gamelift:DescribeGameSessionPlacement",
        "gamelift:StopGameSessionPlacement",
        "gamelift:CreatePlayerSession",
        "gamelift:CreatePlayerSessions",
        "gamelift:DescribeGameSessions"
      ],
      "Resource": "*"
    }
  ]
}

```

Example Sintassi per le autorizzazioni di matchmaking

L'esempio seguente estende l'accesso a coloro Amazon GameLift Servers APIs che gestiscono FlexMatch le attività di matchmaking. FlexMatchabbina i giocatori per sessioni di gioco nuove o esistenti e avvia il posizionamento delle sessioni di gioco per le partite ospitate su. Amazon GameLift Servers Per ulteriori informazioni suFlexMatch, vedi [Cos'è Amazon GameLift ServersFlexMatch?](#)

JSON

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "PlayerPermissionsForGameSessionMatchmaking",
    "Effect": "Allow",
    "Action": [
      "gamelift:StartMatchmaking",
      "gamelift:DescribeMatchmaking",
      "gamelift:StopMatchmaking",
      "gamelift:AcceptMatch",
      "gamelift:StartMatchBackfill",
      "gamelift:DescribeGameSessions"
    ],
    "Resource": "*"
  }
}
```

Example Sintassi per le autorizzazioni di posizionamento manuale delle sessioni di gioco

L'esempio seguente estende l'accesso a coloro Amazon GameLift Servers APIs che creano manualmente sessioni di gioco e sessioni di giocatori su flotte specifiche. Questo scenario supporta giochi che non utilizzano code di posizionamento, ad esempio giochi che consentono ai giocatori di partecipare scegliendo da un elenco di sessioni di gioco disponibili (il metodo "list-and-pick").

JSON

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "PlayerPermissionsForManualGameSessions",
```

```
"Effect": "Allow",
"Action": [
  "gamelift:CreateGameSession",
  "gamelift:DescribeGameSessions",
  "gamelift:SearchGameSessions",
  "gamelift:CreatePlayerSession",
  "gamelift:CreatePlayerSessions",
  "gamelift:DescribePlayerSessions"
],
"Resource": "*"
}
```

Configurare un ruolo di servizio IAM per Amazon GameLift Servers

Alcune Amazon GameLift Servers funzionalità richiedono l'estensione dell'accesso limitato ad altre AWS risorse di tua proprietà. Puoi farlo creando un ruolo AWS Identity and Access Management (IAM). Un [ruolo](#) IAM è un'identità IAM che puoi creare nel tuo account e che dispone di autorizzazioni specifiche. Un ruolo IAM è simile a un utente IAM in quanto è un' AWS identità con politiche di autorizzazione che determinano ciò che l'identità può e non può fare. AWS Tuttavia, invece di essere associato in modo univoco a una persona, un ruolo è destinato a essere assunto da chiunque. Inoltre, un ruolo non ha credenziali a lungo termine standard associate (password o chiavi di accesso). Invece, quando assumi un ruolo, ti fornisce credenziali di sicurezza temporanee per la tua sessione di ruolo.

Questo argomento spiega come creare un ruolo da utilizzare con le flotte Amazon GameLift Servers gestite. [Se lo utilizzi Amazon GameLift ServersFleetIQ per ottimizzare l'hosting di giochi sulle tue istanze Amazon Elastic Compute Cloud \(Amazon EC2\), consulta Configurare il tuo. Account AWSAmazon GameLift ServersFleetIQ](#)

Nella procedura seguente, crea un ruolo con una politica di autorizzazioni personalizzata e una politica di fiducia che Amazon GameLift Servers consenta di assumere il ruolo.

Crea un ruolo di servizio IAM per una flotta Amazon GameLift Servers EC2 gestita

Fase 1: Creare una politica di autorizzazioni.

Utilizza le istruzioni e gli esempi in questa pagina per creare una politica di autorizzazioni personalizzata per il tipo di Amazon GameLift Servers flotta con cui lavori.

Come utilizzare l'editor di policy JSON per creare una policy

1. Accedi Console di gestione AWS e apri la console IAM all'indirizzo <https://console.aws.amazon.com/iam/>.
2. Nel riquadro di navigazione a sinistra, seleziona Policies (Policy).

Se è la prima volta che selezioni Policy, verrà visualizzata la pagina Benvenuto nelle policy gestite. Seleziona Inizia.

3. Nella parte superiore della pagina, scegli Crea policy.
4. Nella sezione Editor di policy, scegli l'opzione JSON.
5. Immettere o incollare un documento di policy JSON. Per dettagli sul linguaggio della policy IAM, consulta la [Documentazione di riferimento delle policy JSON IAM](#).
6. Risolvi eventuali avvisi di sicurezza, errori o avvisi generali generati durante la [convalida delle policy](#), quindi scegli Next (Successivo).

Note

È possibile alternare le opzioni dell'editor Visivo e JSON in qualsiasi momento. Se tuttavia si apportano modifiche o si seleziona Successivo nell'editor Visivo, IAM potrebbe ristrutturare la policy in modo da ottimizzarla per l'editor visivo. Per ulteriori informazioni, consulta [Modifica della struttura delle policy](#) nella Guida per l'utente di IAM.

7. (Facoltativo) Quando crei o modifichi una policy in Console di gestione AWS, puoi generare un modello di policy JSON o YAML da utilizzare nei modelli. CloudFormation

Per fare ciò, nell'editor delle politiche scegli Azioni, quindi scegli Genera modello.

CloudFormation Per saperne di più CloudFormation, consulta il [riferimento al tipo di AWS Identity and Access Management risorsa](#) nella Guida AWS CloudFormation per l'utente.

8. Una volta terminata l'aggiunta delle autorizzazioni alla policy, scegli Successivo.
9. Nella pagina Rivedi e crea, immettere un valore in Nome policy e Descrizione (facoltativo) per la policy in fase di creazione. Rivedi Autorizzazioni definite in questa policy per visualizzare le autorizzazioni concesse dalla policy.
10. (Facoltativo) Aggiungere metadati alla policy collegando i tag come coppie chiave-valore. Per ulteriori informazioni sull'utilizzo dei tag in IAM, consulta [Tags for AWS Identity and Access Management resources](#) in the IAM User Guide.
11. Seleziona Crea policy per salvare la nuova policy.

Fase 2: Crea un ruolo che Amazon GameLift Servers possa assumere.

Per creare un ruolo IAM

1. Nel pannello di navigazione della console IAM, scegliere Ruoli e quindi Crea ruolo.
2. Nella pagina Seleziona entità attendibile, scegli l'opzione Politica di fiducia personalizzata. Questa selezione apre l'editor di criteri di fiducia personalizzati.
3. Sostituisci la sintassi JSON predefinita con la seguente, quindi scegli Avanti per continuare.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "gamelift.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

4. Nella pagina Aggiungi autorizzazioni, individua e seleziona la politica di autorizzazioni creata nel passaggio 1. Seleziona Successivo per continuare.
5. Nella pagina Nome, rivedi e crea, inserisci un nome di ruolo e una descrizione (opzionale) per il ruolo che stai creando. Controlla le entità Trust e le autorizzazioni aggiunte.
6. Scegli Crea ruolo per salvare il tuo nuovo ruolo.

Crea un ruolo IAM per i contenitori Amazon GameLift Servers gestiti

Se utilizzi container Amazon GameLift Servers gestiti, devi creare un ruolo di servizio IAM da utilizzare con una flotta di container. Questo ruolo concede le autorizzazioni limitate Amazon GameLift Servers necessarie per gestire le risorse della flotta di container e intraprendere azioni per tuo conto.

Per creare un ruolo IAM per una flotta di container

1. Accedi Console di gestione AWS e apri la console IAM all'indirizzo <https://console.aws.amazon.com/iam/>.
2. Nel pannello di navigazione della console IAM, scegliere Ruoli e quindi Crea ruolo.
3. Nella pagina Seleziona un'entità affidabile, scegli il AWS servizio e seleziona il caso d'uso "GameLift». Seleziona Next (Successivo).
4. In Aggiungi autorizzazioni, scegli la politica GameLiftContainerFleetPolicy gestita. Scegli Next (Successivo). [AWS politiche gestite per Amazon GameLift Servers](#) Per ulteriori informazioni su questa politica, consulta.
5. In Nome, rivedi e crea, inserisci il nome di un ruolo e scegli Crea ruolo per salvare il nuovo ruolo.

Sintassi della politica di autorizzazione

- Autorizzazioni per Amazon GameLift Servers assumere il ruolo di servizio

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "gamelift.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

- Autorizzazioni per accedere alle AWS regioni che non sono abilitate per impostazione predefinita

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```



```
"Effect": "Allow",
"Principal": {
  "Service": [
    "gamelift.amazonaws.com",
    "gamelift.ap-east-1.amazonaws.com",
    "gamelift.me-south-1.amazonaws.com",
    "gamelift.af-south-1.amazonaws.com",
    "gamelift.eu-south-1.amazonaws.com"
  ]
},
"Action": "sts:AssumeRole"
}
]
```

Ottieni strumenti Amazon GameLift Servers di sviluppo

Amazon GameLift Servers fornisce una serie di SDKs e altri strumenti per aiutarti a creare soluzioni di hosting di giochi per i tuoi giochi. SDKs Aggiungono funzionalità ai server di gioco, ai client di gioco e ai servizi di backend che consentono loro di interagire con il Amazon GameLift Servers servizio. Per le informazioni più recenti sulle versioni e sulla compatibilità dell'Amazon GameLift Servers SDK, consulta. [Note di rilascio di Amazon GameLift Servers](#)

Per i server di gioco

Integra e crea i tuoi server di gioco a 64 bit con il server SDK for Amazon GameLift Servers. Il server di gioco utilizza l'SDK del server per comunicare con il Amazon GameLift Servers servizio per la gestione delle sessioni di gioco, inclusi l'avvio, l'aggiornamento e l'interruzione delle sessioni di gioco. Per assistenza sull'integrazione dell'SDK del server nei tuoi progetti di gioco, consulta. [Prepara un gioco da ospitare con Amazon GameLift Servers](#)

Supporto allo sviluppo

- Sistema operativo di sviluppo
 - Windows
 - Linux
- Linguaggi di programmazione

[Scarica l'Amazon GameLift Servers SDK](#). Per informazioni specifiche sulla versione e istruzioni di installazione, consultate i file readme inclusi in ogni pacchetto.

- [SDK per server C++](#)
 - [Riferimento all'SDK del server](#)
 - [Come effettuare l'integrazione](#)
- [SDK per server C#](#) (il supporto per .NET 4, .NET 6, .NET 8 varia in base alla versione, vedi [Versioni SDK](#))
 - [Riferimento all'SDK del server](#)
 - [Come effettuare l'integrazione](#)
- [Vai al server SDK](#)
 - [Riferimento all'SDK del server](#)
 - [Come effettuare l'integrazione](#)
- Supporto per motori di gioco

Il plug-in completo Amazon GameLift Servers include flussi di lavoro dell'interfaccia utente e risorse di esempio, oltre a versioni integrate dell' AWS SDK e dell'SDK del server. I flussi di lavoro ti guidano attraverso come configurare e implementare il tuo server di gioco per l'hosting con flotte EC2 gestite, flotte di container gestite o flotte Anywhere autogestite. Se non hai bisogno dei flussi di lavoro guidati, puoi scaricare solo l'SDK del server per il tuo motore di gioco dagli stessi repository. GitHub

Se utilizzi un altro motore di gioco o un altro ambiente di sviluppo non supportato dal plug-in, scarica l'SDK del server per il tuo linguaggio di programmazione e aggiungilo al tuo progetto di gioco.

Per informazioni specifiche sulla versione e istruzioni di installazione, consulta i file readme inclusi in ogni pacchetto.

- [Plugin per Unreal Engine](#): creato per l'uso con le versioni 5.6. Controlla i file readme specifici della versione per il supporto di Unreal.
 - [Guida ai plugin per Unreal Engine](#)
 - [Server C++ \(Unreal\) SDK 5.x per -- Azioni Amazon GameLift Servers](#)
- [Plugin per Unity](#): creato per l'uso con le versioni LTS di Unity Editor 6.0, 2022.3 o 2021.3. Supporta i profili .NET Framework e .NET Standard di Unity, con .NET Standard 2.1 e .NET 4.x. Controlla i file readme specifici della versione per il supporto di Unity.

- [Guida ai plugin per Unity](#)
- [Riferimento all'SDK del server C#](#)
- [Server SDK per Unreal](#)
 - [Riferimento all'SDK del server](#)
 - [Integrazione Amazon GameLift Servers in un progetto Unreal Engine](#)
- [SDK per server per Unity](#)
 - [Riferimento all'SDK del server](#)
 - [Integrazione Amazon GameLift Servers in un progetto Unity](#)

Supporto in fase di esecuzione

Per una soluzione di hosting gestito, crea il tuo server di gioco in modo che funzioni su una delle seguenti immagini di macchine Amazon (AMIs). [Amazon GameLift Servers Versioni AMI](#) Amazon GameLift Servers Per ulteriori dettagli, consulta la pagina.

- [Windows Server 2016](#)
- [Amazon Linux 2023](#)
- [Amazon Linux 2](#)

Note

Amazon Linux 2 (AL2) terminerà il supporto il 30 giugno 2026. Scopri maggiori dettagli in [Amazon Linux 2 FAQs](#). Per i server di gioco ospitati AL2 e che utilizzano il Amazon GameLift Servers server SDK 4.x, aggiorna prima la build del server di gioco su server SDK 5.x, quindi distribuiscila sulle istanze. AL2023 Per informazioni, consulta [Esegui la migrazione al server SDK 5.x per Amazon GameLift Servers](#).

Strumenti aggiuntivi

[Wrapper per server di gioco per Amazon GameLift Servers](#)

Questo strumento ti aiuta a implementare un server di gioco per l'hosting con una serie di funzionalità di base per la gestione delle sessioni di gioco. Con questo strumento, non è necessario apportare modifiche al codice di gioco o integrare l'SDK del server per. Amazon GameLift Servers Usa il

wrapper del server di gioco per impacchettare il tuo server di gioco e distribuirlo per l'hosting di giochi utilizzando una delle tre soluzioni di Amazon GameLift Servers hosting (Anywhere, EC2 gestito o container gestiti). Questo strumento è ideale per la valutazione precoce o la creazione di prototipi con il tuo gioco o con un gioco di esempio, poiché non supporta la personalizzazione del server di gioco. Se il tuo gioco non necessita di funzionalità personalizzate, puoi utilizzare il server di gioco con il wrapper del server di gioco per l'hosting di produzione.

[Amazon GameLift ServersKit di strumenti](#)

Il Amazon GameLift Servers Toolkit è una raccolta di script e altri strumenti che abbiamo sviluppato per aiutare gli sviluppatori con scenari e problemi comuni. I materiali del toolkit includono script, codice di esempio e file readme.

- [Containers Starter Kit](#): utilizza questo strumento per semplificare le attività di configurazione delle build di server di gioco per l'hosting con contenitori gestiti. Amazon GameLift Servers Il kit integra le funzionalità essenziali di gestione delle sessioni di gioco in un server di gioco e utilizza modelli preconfigurati per creare una flotta di container e impostare una pipeline di distribuzione automatizzata per la creazione del server di gioco. Dopo l'implementazione, puoi monitorare le prestazioni della flotta, gestire le sessioni di gioco e analizzare le metriche utilizzando la Amazon GameLift Servers console e gli strumenti API. Il kit si integra con Amazon Simple Storage Service AWS CodeBuild per l'automazione delle build, per lo storage e AWS CloudFormation per l'implementazione dell'infrastruttura.
- [Strumento di aggiornamento rapido della build](#): utilizza questo strumento per modificare una build di server di gioco già implementata in una flotta EC2 gestita. Lo strumento è progettato per aiutarti a sostituire rapidamente i file di build del gioco senza dover configurare e creare nuove flotte EC2 ad ogni modifica. Puoi aggiornare singole istanze o tutte le istanze del parco istanze. Le opzioni ti consentono di sostituire file di build specifici o un'intera build e ti consentono di gestire il riavvio dei server di gioco dopo gli aggiornamenti.

Per i client di gioco

Integra il tuo client di gioco con il client SDK forAmazon GameLift Servers. L'SDK del client fornisce utilità per consentire ai client di gioco di utilizzare Amazon GameLift Servers funzionalità tra cui player gateway e ping beacon UDP.

Scarica l'SDK del client

- [Client SDK for C++](#): include le utilità player gateway e ping beacon UDP con algoritmi di selezione degli endpoint integrati, oltre a un'applicazione di esempio che dimostra l'integrazione. Richiede C++17 e una libreria di threading.
- [Client SDK per Unreal Engine: include player gateway \(con o senza modifiche al motore\)](#) e utilità di ping beacon UDP con algoritmi di selezione degli endpoint integrati. Richiede Unreal Engine 5.1 o versione successiva integrata nel codice sorgente. L'accesso richiede l'iscrizione all'organizzazione [Epic Games](#) il. GitHub

Per le guide all'integrazione, consulta la documentazione in ogni GitHub repository.

Per i servizi di backend dei giochi

Crea un servizio di backend a 64 bit per il tuo gioco e integralo con le funzionalità dell' AWS SDK, che include l'API di servizio per. Amazon GameLift Servers Utilizza il servizio di backend per gestire le interazioni con il Amazon GameLift Servers servizio, tra cui l'avvio o la ricerca di sessioni di gioco e l'aggiunta di giocatori alle partite.

[Scarica l'SDK AWS](#)

Per ulteriori informazioni sull'utilizzo dell' AWS SDK con Amazon GameLift Servers, consulta le seguenti risorse:

- [Documentazione di riferimento dell'API Amazon GameLift Servers](#)
- Integra la funzionalità del client di gioco
 - [Integrazione dei servizi lato client per giochi con build di server di gioco personalizzate](#)
 - [Integrazione dei servizi lato client per giochi con server Realtime](#)
- [Progetta un servizio di backend](#)

Per la Amazon GameLift Servers gestione delle risorse

Utilizza i seguenti strumenti per creare, aggiornare e monitorare le tue risorse di hosting Amazon GameLift Servers gestito.

- [Console di gestione AWS](#)— La AWS console è un'applicazione basata sul Web che fornisce l'accesso centralizzato a tutte le singole console di AWS servizio, tra cui. Amazon GameLift Servers Usa la console per creare o accedere a un AWS account e aprirla per utilizzare le Amazon

GameLift Servers risorse di hosting dei giochi. Puoi configurare e distribuire flotte di hosting e altre risorse, visualizzare le metriche di utilizzo e prestazioni, tenere traccia delle risorse nella dashboard e molte altre attività. [Vai alla console. Amazon GameLift Servers](#)

- [API di servizio per Amazon GameLift Servers](#): questa API ti offre l'accesso programmatico a tutte le tue Amazon GameLift Servers risorse. Fa parte dell' AWS SDK, che puoi scaricare per utilizzarlo con i linguaggi di programmazione più diffusi. [Scarica l' AWS SDK](#).
- [AWS interfaccia a riga di comando \(CLI\)](#): la AWS CLI consente di interagire con i AWS servizi utilizzando una shell a riga di comando. Gli strumenti forniscono l'accesso diretto al pubblico ai AWS servizi e ai comandi personalizzati disponibili APIs per un servizio. [Scarica la AWS CLI](#).
- [AWS CloudFormation](#) per Amazon GameLift Servers: Il CloudFormation servizio ti aiuta a modellare e configurare AWS le risorse per semplificare l'implementazione e la gestione dell'infrastruttura. Crea un CloudFormation modello per descrivere le Amazon GameLift Servers risorse per la tua soluzione di hosting, quindi utilizza il modello per creare risorse aggiuntive o aggiornare le configurazioni. Visualizza il [riferimento al tipo di Amazon GameLift Servers risorsa](#).

Per Amazon GameLift Servers Realtime

Configura e distribuisce Realtime server per ospitare le tue partite multiplayer. Per consentire ai client di gioco di connettersi ai Realtime server, utilizza l'SDK del Amazon GameLift Servers Realtime client. Per iniziare, [scarica l'SDK del Realtime client](#). Per informazioni sulla configurazione, consulta [Integrazione di un client di gioco](#) per. Amazon GameLift Servers Realtime

Supporto SDK

Il client SDK Realtime contiene la sorgente per i seguenti linguaggi:

- C# (.NET)

Ambienti di sviluppo

Crea l'SDK dal codice sorgente in base alle esigenze per i seguenti sistemi operativi di sviluppo e motori di gioco supportati:

- Sistemi operativi: Windows, Linux, Android, iOS
- Motori di gioco: Unity, motori che supportano le librerie C#
- Runtime di Node.js: 10.x, 24.x

 Note

Il supporto per Node.js 10 terminerà il 30 settembre 2026. Per maggiori dettagli, consulta [Node.js 10 FAQs](#). Per indicazioni sulla migrazione, vedi [Migrazione da Node.js 10 a 24](#).

Sistemi operativi dei server di gioco

È possibile distribuire Realtime server su risorse di hosting eseguite sulle seguenti piattaforme:

- [Amazon Linux 2023](#)
- [Amazon Linux 2](#)

Tutorial: onboarding rapido con il wrapper Amazon GameLift Servers

Benvenuto nel tutorial di onboarding per Amazon GameLift Servers. In questo tutorial, implementerai rapidamente il tuo server di gioco in modo che sia ospitato su una flotta di risorse di calcolo basate sul cloud. Usa questo tutorial per evitare di integrare l'SDK del server Amazon GameLift Servers nel codice del gioco e distribuirlo invece il gioco con le funzionalità minime necessarie per comunicare con il servizio ed eseguire sessioni di gioco. Amazon GameLift Servers Configurerai una soluzione di hosting di base e la utilizzerai per provare l'intera gamma di funzionalità come il ridimensionamento automatico e il matchmaking. È anche un ottimo modo per ospitare un prototipo del tuo gioco come parte di una demo dal vivo o per un test.

Principali vantaggi di questo metodo di onboarding:

- Implementa rapidamente il tuo server di gioco per un hosting rapido.
- Zero modifiche al codice di gioco e nessuna modifica richiesta.
- Usa questo metodo con qualsiasi file eseguibile di gioco, indipendentemente dal motore di gioco.
- Esplora tutti gli strumenti di Amazon GameLift Servers gestione, incluso il monitoraggio dell'attività delle sessioni di gioco e lo stato dell'hosting.

 Note

Il wrapper è destinato alla valutazione e all'uso di base nella produzione. Le funzionalità avanzate, come la gestione dettagliata delle sessioni dei giocatori, richiedono un'integrazione completa dell'SDK del server.

Prerequisiti

Prima di iniziare, assicurati di disporre dei seguenti elementi:

- E Account AWS con le autorizzazioni appropriate
- AWS CLI installato
- Vai a 1.18+
- Un server di gioco multiplayer eseguibile
- Crea (Linux/Mac)
- Git installato con un account attivo

Panoramica di

In questo tutorial potrai:

1. Prendi e costruisci l'involucro
2. Prepara la build del gioco
3. Configura il wrapper
4. Carica la build del server di gioco
5. Crea la flotta EC2 gestita
6. Crea e connettiti a una sessione di gioco
7. Monitora e gestisci i tuoi server di gioco
8. Ridimensiona i tuoi server di gioco

Passaggio 1: Ottieni e crea il wrapper del server di gioco

Usa i seguenti comandi per ottenere il sorgente del wrapper del server di gioco e creare il wrapper. Questi comandi usano SSH, ma puoi anche accedere direttamente al repository. GitHub

Windows

```
> git clone git@github.com:amazon-gamelift/amazon-gamelift-servers-game-server-wrapper.git
> cd amazon-gamelift-servers-game-server-wrapper
> powershell -file .\build.ps1
```

Mac e Linux

```
$ git clone git@github.com:amazon-gamelift/amazon-gamelift-servers-game-server-wrapper.git
$ cd amazon-gamelift-servers-game-server-wrapper
$ make
```

In caso di compilazione riuscita, viene aggiunta una directory «out» a `amazon-gamelift-servers-game-server-wrapper`. In questa directory ci sono tre cartelle, una per ogni opzione della flotta di hosting supportata, che contengono un set di elementi di build. Per questo tutorial, eseguirai l'implementazione su flotte EC2 gestite, quindi utilizzerai la cartella `gamelift-servers-managed-ec2`.

Passaggio 2: prepara la build del server di gioco

In questo passaggio, prepari i file di build del server di gioco da caricare su Amazon GameLift Servers.

Crea la directory del gioco

Ora prepara una directory di gioco sul tuo computer locale. Questa directory deve contenere tutti i file necessari per eseguire il server di gioco Amazon GameLift Servers. Ciò include il wrapper del server di gioco, la build del server di gioco e il file `config.yaml` che fa funzionare il wrapper con il server di gioco.

Utilizza le fasi seguenti:

1. Flotta EC2 gestita. Nella cartella di output del wrapper del server di gioco, trova l'artefatto di build da distribuire in una flotta EC2 gestita. La build verrà scritta in una directory esterna come indicato qui: `out\linux\amd64\gamelift-servers-managed-ec2`
2. Copia il file eseguibile del server di gioco e tutti i file associati necessari per essere eseguiti nella `gamelift-servers-managed-ec2` cartella. Se necessario, puoi avere cartelle annidate.

Un esempio di struttura di cartelle sarà simile a questo:

```
gamelift-servers-managed-ec2
#-- config.yaml
#-- amazon-gamelift-servers-game-server-wrapper
#-- MyGame
#   #-- server-executable.exe
#   #-- my-game-settings
#   # .....
```

Fase 3: Configura il wrapper per la tua flotta

Amazon GameLift Servers gestisce il ciclo di vita delle istanze di calcolo di una flotta, creando nuove istanze con la build del server installata e riciclando le istanze secondo necessità. Il servizio gestisce il ciclo di vita dei processi del server di gioco che viene eseguito su ogni istanza. Una flotta EC2 gestita può avere istanze in più sedi per supportare i giocatori ovunque si trovino.

Modifica il file `config.yaml` per configurare il wrapper per la registrazione, la configurazione delle porte e l'inizializzazione del server.

1. Configura le impostazioni di registrazione. Il wrapper del server di gioco genera messaggi di registro per ogni processo del server di gioco. Per impostazione predefinita, il livello di registro è impostato su debug per la massima verbosità. Ciò è molto utile durante la configurazione e la risoluzione dei problemi e determina il livello di dettaglio dei messaggi di registro, in questo caso i più dettagliati. Le opzioni includono debug, info, warn ed error (meno dettagliato).
2. Specificate il percorso della directory di registro del server di gioco. Il percorso predefinito per i log del server di gioco è `./game-server-logs`. Questa directory contiene tutti i log generati dal server di gioco e ogni istanza li contiene. I log vengono caricati automaticamente nel Amazon GameLift Servers punto in cui è possibile accedervi dalla scheda Eventi. Visualizza la sezione Risoluzione dei problemi per maggiori dettagli.

3. Definire la configurazione delle porte di rete. Imposta la porta di gioco come preferisci. Per questo tutorial, specifica una sola porta, poiché creerai una flotta che esegue solo un processo simultaneo del server di gioco per istanza. Se decidi di eseguire più processi contemporaneamente, dovrai configurare un numero sufficiente di porte per ogni processo simultaneo. Il valore predefinito è 37016, come mostrato nel file di configurazione, ma in generale, per le flotte che utilizzano build Linux, utilizza le porte 22 e 1026-60000. Per le flotte che utilizzano build Windows, utilizzate le porte 1026-60000.
4. Imposta il percorso dell'eseguibile del server di gioco. Per `./MyGame/my-server-executable.exe` personalizzare il percorso dell'eseguibile del server di gioco con il nome e la posizione effettivi. Questo è il punto di accesso per avviare il server di gioco.
5. Configura gli argomenti del server di gioco. Come minimo, specifica un argomento `--port` e usa lo stesso valore di porta di gioco che hai definito in precedenza. Il valore «pos» 0 indica che questo è il primo argomento. Aggiungi altri argomenti se necessario. Questi argomenti vengono passati al server di gioco all'avvio, consentendoti di configurarne il comportamento in fase di esecuzione.
 - a. Argomento: `--port`
 - b. Valore: `"{{.port number here}}"`
 - c. Posizione: 0 (primo argomento nell'elenco)

Configurazione di esempio:

```
log-config:
  wrapper-log-level: debug
game-server-logs-dir: ./game-server-logs
ports:
  gamePort: 37016
game-server-details:
  executable-file-path: ./MyGame/my-server-executable
game-server-args:
  - arg: "--port"
    val: "{{.gameport}}"
    pos: 0
```

Passaggio 4: carica la build del server di gioco

Ora hai completato tutti gli elementi richiesti della build del tuo server di gioco (game server wrapper, config.yaml e i file del server di gioco) e sei pronto per caricare la build del gioco su come hosting. Amazon GameLift Servers Il modo più rapido per caricare la build del gioco è usare il AWS CLI comando, come mostrato nell'esempio seguente `upload-build`.

Caricare una build di gioco con Windows:

```
aws gamelift upload-build \  
  --name gamelift-test-2025-03-11-1 \  
  --build-version gamelift-test-2025-03-11-1 \  
  --build-root out/windows/amd64/gamelift-servers-managed-ec2 \  
  --operating-system WINDOWS_2022 \  
  --server-sdk-version 5.3.0 \  
  --region us-west-2
```

Note

Per le versioni Mac e Linux, usa `--operating-system AMAZON_LINUX_2023`

Quando crei la build, registra l'ID di build dalla risposta dell'API per utilizzarlo per la creazione della flotta.

Fase 5: Crea la flotta EC2 gestita

I passaggi seguenti descrivono una configurazione minima della flotta, in modo da poter essere operativi il prima possibile.

Per creare la tua flotta:

1. Accedi a Console di gestione AWS e vai a Amazon GameLift Servers.
2. Nella barra dei menu nella parte superiore della finestra della console, verifica in quale regione si trova la build. Prendine nota perché la tua flotta deve trovarsi nella stessa regione o non potrai trovare o scegliere la tua build.
3. Nel pannello di navigazione della sezione Managed EC2, scegli Builds.

4. Seleziona la build che hai caricato in precedenza per visualizzare la pagina dei dettagli della build.
5. Nella sezione Flotte, scegli Crea flotta, che visualizza la pagina dei dettagli della flotta Define Managed EC2, da cui puoi monitorare lo stato della tua flotta e visualizzare gli eventi di creazione della flotta nella scheda Eventi.
6. Compila il nome e la descrizione e scegli Avanti.
7. Nella pagina Definisci i dettagli dell'istanza, l'area della build viene mostrata per impostazione predefinita. Scegli le aree aggiuntive che desideri aggiungere.
8. Per Tipo di flotta scegli On-Demand.
9. In Tipi di istanze scegli c5.large e scegli Avanti.
10. Nella configurazione Runtime, poiché la build del gioco caricata utilizza il wrapper, devi invece puntare all'eseguibile wrapper. Per i server di gioco Windows questo è. `C:\game\amazon-gamelift-servers-game-server-wrapper.exe` Per i server di gioco Linux, questo è `/local/game/amazon-gamelift-servers-game-server-wrapper`.

Ad esempio: `LaunchPath": "/local/game/amazon-gamelift-servers-game-server-wrapper", "ConcurrentExecutions": 1, "Parameters": "-port 37016`

Inoltre, configura i valori della porta di gioco con un intervallo che tenga conto della porta impostata nei `config.yaml` parametri di avvio della configurazione di runtime. `config.yaml` Non è necessario che la porta in ingresso corrisponda a quella specificata nella configurazione di runtime della flotta, ma in fase di esecuzione se la configurazione specifica una porta diversa, quel valore prevale su quello contenuto in `config.yaml`. I parametri di avvio immessi nella configurazione di runtime hanno inoltre la precedenza su ciò che è contenuto in `config.yaml`.

11. Nella pagina Rivedi e crea, ricontrolla tutte le configurazioni, quindi scegli Invia per creare la tua flotta. Lo stato della flotta cambierà man mano che aumenterà la capacità necessaria per ospitare il server di gioco e molto presto mostrerà lo stato Attivo. Una volta completata l'attivazione e schierata la flotta, il servizio avvia il wrapper, pronto a ricevere una richiesta di sessione di gioco.

Passaggio 6: Crea e connettiti a una sessione di gioco

Quando lo stato della tua flotta risulta Attivo, significa che i server di gioco sono pronti e in attesa di ospitare una sessione di gioco. Per iniziare una sessione di gioco, invia una richiesta di sessione di gioco al Amazon GameLift Servers servizio. Qui, utilizzerai il AWS CLI per effettuare questa richiesta.

 Note

Tieni presente che creare una sessione di gioco utilizzando il AWS CLI è utile per testare e acquisire familiarità con il processo. Ad un certo punto, aggiungerai chiamate AWS SDK programmatiche al servizio di backend di gioco come parte del sistema di matchmaking o di posizionamento delle sessioni di gioco.

Usa quanto segue per creare una sessione di gioco:

```
aws gamelift create-game-session \  
--fleet-id <FLEET_ID> \  
--game-properties '[{"Key": "exampleProperty", "Value": "exampleValue"}]' \  
--maximum-player-session-count 3 \  
--region us-west-2
```

Puoi anche passare proprietà di gioco personalizzate all'eseguibile del tuo server. Vedi gli argomenti del server di gioco nel README per i dettagli. Quando riceve la create-game-session chiamata, Amazon GameLift Servers informa il wrapper di avviare l'eseguibile del server di gioco e avviare una sessione di gioco. Il contenuto `config.yaml` influisce sulla configurazione del server di gioco e i parametri di avvio impostati nella console determinano la configurazione della sessione di gioco stessa.

Formato di esempio per aggiungere le proprietà del gioco:

```
defaultArgs:  
  - arg: "--port"  
    val: "{{.GamePort}}"  
    pos: 0  
  - arg: "--ipAddress"  
    val: "{{.IpAddress}}"  
    pos: 1  
  - arg: "--gameSessionId"  
    val: "{{.GameSessionId}}"  
    pos: 2
```

Fase 7: Gestisci e monitora la tua flotta

Ora che il tuo parco server di gioco è configurato e la sessione di gioco è iniziata, puoi gestirla e monitorarla dalla Amazon GameLift Servers console. Il modo migliore per farlo è accedere alla pagina dei dettagli della flotta, dove puoi modificare i dettagli della flotta o modificare la scalabilità e la capacità della flotta nella scheda Scaling. Consulta la sezione seguente sul ridimensionamento dei server di gioco.

Scegli la scheda Metriche per visualizzare i grafici che illustrano il monitoraggio delle metriche relative all'attività e all'hardware. Per i dettagli sui grafici delle metriche, scegli il link Informazioni accanto all'ID del tuo parco veicoli. Inoltre, puoi monitorare da vicino i tuoi server di gioco dalla scheda Metriche, ma puoi anche aggiungere allarmi a queste metriche nella dashboard. CloudWatch

Per accedere alla CloudWatch dashboard dalla console:

1. Digita CloudWatch "" nella barra di ricerca e selezionalo dall'elenco dei risultati della ricerca per visualizzare la CloudWatch panoramica.
2. Scorri verso il basso e scegli Visualizza GameLift dashboard per visualizzare i grafici basati sulle metriche chiave per le tue flotte e le sessioni di gioco.

Passaggio 8: ridimensiona i server di gioco

Per il passaggio successivo, configuri il ridimensionamento automatico. Con la scalabilità automatica puoi scalare dinamicamente la capacità della tua flotta in risposta all'attività dei server di gioco. Man mano che i giocatori arrivano e iniziano le sessioni di gioco, la scalabilità automatica aggiunge altre istanze e, man mano che la domanda dei giocatori diminuisce, la scalabilità automatica rilascia le istanze inutilizzate. Si tratta di un modo efficace per ridurre al minimo le risorse e i costi di hosting, garantendo al contempo un'esperienza di gioco fluida e veloce.

Mentre ti prepari per il lancio del gioco, ti consigliamo di impostare la scalabilità automatica per le tue flotte. La scalabilità automatica è consigliata come metodo efficace per ridurre al minimo le risorse e i costi di hosting, garantendo al contempo un'esperienza di gioco fluida e veloce.

Per impostare manualmente la capacità del parco veicoli

Per impostare manualmente la capacità del parco veicoli

1. Vai alla scheda Scaling nella pagina dei dettagli della tua flotta.
2. Seleziona una località e scegli Modifica.

3. Modifica il valore delle istanze desiderate e le impostazioni delle dimensioni minime e massime per scalarle oltre i valori correnti, quindi scegli Conferma.

Note

Utilizza l'impostazione del numero massimo di istanze come intervallo temporale per evitare costi e scalabilità eccessivi.

Per utilizzare il ridimensionamento automatico basato sulla destinazione

Per utilizzare il ridimensionamento automatico basato sulla destinazione

La scalabilità automatica basata sugli obiettivi (tracciamento degli obiettivi) collega il ridimensionamento della flotta alla percentuale di sessioni di gioco disponibili. Man mano che i giocatori si affollano a giocare al tuo gioco e le sessioni di gioco disponibili diminuiscono, il sistema risponde aggiungendo automaticamente altre istanze al parco istanze.

1. In Politica di scalabilità automatica basata su Target, scegli Aggiungi criterio e imposta la capacità della flotta in modo che cambi automaticamente quando raggiunge la soglia della percentuale di sessioni di gioco disponibili che hai impostato. Un buffer più ampio può gestire meglio le sovratensioni, far sì che i nuovi giocatori entrino rapidamente nei giochi, ma può anche comportare costi di hosting più elevati.
2. Scegli Conferma per accettare le modifiche.

La scalabilità automatica basata su regole offre un controllo più granulare, ad esempio la possibilità di collegare la scalabilità ad altre metriche della flotta e di impostare soglie e risposte di scalabilità personalizzate. Offre opzioni potenti, ma richiede anche l'utilizzo della CLI e test significativi per capire come si comportano le regole personalizzate in azione. Questo tutorial si concentra sulla configurazione iniziale dell'approccio basato sugli obiettivi.

Risolvi i problemi più comuni

Di seguito sono riportati i problemi più comuni che potresti riscontrare durante l'esecuzione dei server e delle sessioni di gioco. Quando il server o la sessione di gioco non funzionano correttamente, la prima cosa da fare è controllare i log, che potrebbero rivelare uno dei problemi descritti di seguito per le nuove distribuzioni o i giochi in produzione.

Nei log viene spesso rivelato quanto segue:

- Il processo del server di gioco non può essere avviato. Potrebbe trattarsi di un errore nella configurazione del wrapper: verifica che il file abbia il percorso di avvio corretto e i parametri e gli argomenti di avvio corretti.
- La build del server di gioco non è eseguibile. È probabile che si tratti di un errore nel codice del gioco.
- I giocatori non possono connettersi alle sessioni di gioco. È probabile che si tratti di un errore di configurazione della porta.
- Connessioni lente o in ritardo. Rivedi le politiche e le soglie di scalabilità.
- Nessuna connessione. Verifica le regole e la configurazione dei porti per la tua flotta.

Per visualizzare i registri degli eventi della tua flotta Amazon GameLift Servers

Per visualizzare i registri degli eventi della tua flotta Amazon GameLift Servers

1. Aprire la console Amazon GameLift Servers.
2. Nella pagina Dettagli della flotta, scegli la scheda Eventi e scarica il registro. Puoi anche monitorare le attività e le metriche hardware relative allo stato del server di gioco e alle attivazioni delle sessioni di gioco dalla scheda Metriche.

Per visualizzare i registri delle sessioni di gioco

Per visualizzare i registri delle sessioni di gioco

1. Dalla console, apri la tua flotta e apri la scheda Sessioni di gioco.
2. Scegli un ID della sessione di gioco dall'elenco per visualizzare la relativa pagina Panoramica.
3. Scegli Scarica i registri per scaricare un file di registro localmente.

Per visualizzare i registri delle sessioni di gioco con la CLI, utilizza l'`GetGameSessionLogURLAPI`. Amazon GameLift Servers archivia automaticamente i log per 14 giorni.

Puoi anche configurare CloudWatch i log di Amazon per la tua flotta. Ciò fornisce funzionalità di registrazione aggiuntive e integrazione con altri servizi di AWS monitoraggio.

Per l'accesso ai log in tempo reale o per periodi di conservazione prolungati tramite CloudWatch:

1. Nella parte superiore della dashboard della Amazon GameLift Servers console, digita "CloudWatch" nella barra di ricerca e selezionala dal menu a discesa dei risultati.
2. Vai ai gruppi di CloudWatch log e cerca sessioni specifiche. Il metodo più semplice è fare clic su Cerca tutto e filtrare utilizzando gameSessionId o ClientID.

Fasi successive

- [Crea una flotta con più sedi per aggiungere hosting in altre sedi](#)
- [Aggiungi una coda per le sessioni di gioco per ottenere i migliori posizionamenti possibili per le sessioni di gioco in più sedi](#)
- [Sperimenta con il FlexMatch matchmaking creando un matchmaker e un set di regole per il tuo gioco](#)
- [Inizia a lavorare sulle funzionalità del client di gioco e dei componenti del servizio di backend, in modo che i giocatori possano effettuare richieste di iscrizione e connettersi direttamente alle sessioni di gioco](#)
- [Quando sei pronto, passa a una soluzione completamente integrata](#)

Esempi di Amazon GameLift Servers

Se stai pensando di Amazon GameLift Servers utilizzarlo per gestire il tuo server di gioco personalizzato o sei interessato a sfruttarlo Amazon GameLift ServersRealtime, ti consigliamo di provare i seguenti esempi prima di utilizzare il servizio per il tuo gioco. L'esempio di server di gioco personalizzato ti offre un'esperienza con l'hosting di giochi sulla Amazon GameLift Servers console. L'Amazon GameLift ServersRealtimeesempio mostra come preparare un gioco per l'hosting tramite Realtime server.

Esempio di server di gioco personalizzato

Questo esempio dimostra il processo di implementazione di un server di gioco di esempio su una EC2 flotta Amazon GameLift Servers gestita per l'hosting. Usa il client di gioco di esempio per connetterti a una sessione di gioco dal vivo. Puoi scoprire come utilizzare Amazon GameLift Servers .tools, tra cui la console e la AWS CLI, per monitorare le prestazioni e l'utilizzo dell'hosting della flotta.

L'esempio illustra i seguenti passaggi:

- Carica la build di esempio del server di gioco.
- Crea una flotta per eseguire la build del server di gioco.
- Scarica il client di gioco di esempio e usalo per connetterti a un server di gioco e partecipare a una sessione di gioco.
- Rivedi le metriche della flotta e delle sessioni di gioco.

Avvia più client di gioco e gioca per generare dati di hosting. Usa la Amazon GameLift Servers console per visualizzare le risorse di hosting, tenere traccia delle metriche ed esplorare le opzioni per scalare la capacità di hosting della flotta.

[Per iniziare, accedi alla Amazon GameLift Servers console.](#) Nella barra di navigazione a sinistra, vai su Risorse, Prova un gioco di esempio.

Creazione di una soluzione di hosting di giochi con Amazon GameLift Servers

Questa sezione include indicazioni su come creare tutti i componenti di una soluzione di hosting di Amazon GameLift Servers giochi, comprese le migliori pratiche di progettazione. Che tu stia iniziando da zero o migrando un gioco esistente, troverai informazioni complete per aiutarti a progettare, implementare e ottimizzare la tua infrastruttura di hosting di giochi utilizzando Amazon GameLift Servers.

Componenti della soluzione di hosting

La tua soluzione di hosting deve avere una versione dei seguenti componenti. Per ulteriori dettagli su questi componenti e sul modo in cui interagiscono, consulta [Come Amazon GameLift Servers funziona l'hosting con](#).

- Il client di gioco funziona sui dispositivi dei giocatori.
- Il servizio di backend coordina la comunicazione tra i client di gioco e Amazon GameLift Servers.
- Il software del server di gioco gestisce il gameplay multiplayer.
- Il sistema di posizionamento abbina i giocatori ai server di gioco.
- Le flotte di hosting di giochi forniscono risorse informatiche in posizioni geografiche.
- Il sistema di gestione dell'hosting di giochi monitora lo stato di hosting dei giochi e gestisce la capacità.

Argomenti

- [Roadmap di sviluppo](#)
- [Prepara un gioco da ospitare con Amazon GameLift Servers](#)
- [Crea un servizio di backend per Amazon GameLift Servers](#)
- [Implementa flotte di hosting per Amazon GameLift Servers](#)
- [Configura il posizionamento delle sessioni di gioco](#)
- [Personalizzalo in base alla tua soluzione di hosting di giochi](#)

Roadmap di sviluppo

Le roadmap di sviluppo forniscono step-by-step una guida strutturata per la creazione di diversi tipi di soluzioni di hosting di giochi con Amazon GameLift Servers. Ogni tabella di marcia è personalizzata in base a un approccio di hosting specifico e illustra le attività essenziali, dalla configurazione iniziale all'implementazione e al test.

Scegli la roadmap più adatta ai tuoi requisiti di hosting e ai tuoi vincoli tecnici. Puoi anche combinare gli approcci seguendo diverse tabelle di marcia per creare una soluzione ibrida che soddisfi le tue esigenze specifiche.

Argomenti

- [Roadmap di sviluppo per l'hosting con EC2 Amazon GameLift Servers gestito](#)
- [Roadmap di sviluppo per l'hosting con contenitori Amazon GameLift Servers gestiti](#)
- [Roadmap di sviluppo per l'hosting con Anywhere Amazon GameLift Servers](#)
- [Roadmap di sviluppo per l'hosting ibrido con Amazon GameLift Servers](#)

Roadmap di sviluppo per l'hosting con EC2 Amazon GameLift Servers gestito

Questa tabella di marcia ti guida nello sviluppo di una soluzione di hosting EC2 Amazon GameLift Servers gestita per il tuo gioco multiplayer. Amazon GameLift Servers offre diverse opzioni di hosting di giochi; per ulteriori informazioni su queste opzioni, consulta [Amazon GameLift Servers opzioni di hosting di giochi](#)

Con l'hosting Amazon GameLift Servers gestito, il server di gioco è ospitato su risorse informatiche virtuali Cloud AWS basate su risorse di calcolo virtuali che Amazon GameLift Servers possiedono e operano in base alla configurazione dell'utente. Ottieni la sicurezza, l'affidabilità e la disponibilità globale delle istanze Amazon Elastic Compute Cloud (Amazon EC2) ulteriormente ottimizzate per l'uso con l'hosting di giochi multiplayer. Amazon GameLift Servers semplifica la gestione dell'hosting con strumenti come l'implementazione automatica dei server, la gestione del ciclo di vita e la scalabilità automatica della capacità.

Una soluzione Amazon GameLift Servers gestita è composta dai seguenti componenti:

- Una versione di server di gioco, integrata con l'SDK del server per Amazon GameLift Servers, da distribuire su tutte le flotte.

- Un client di gioco e un servizio di backend, integrato con l' AWS SDK, per interagire con il Amazon GameLift Servers servizio e richiedere sessioni di gioco.
- Una Amazon GameLift Servers coda o un altro meccanismo di posizionamento per inserire nuove sessioni di gioco con i server di gioco disponibili in tutte le flotte.
- (Facoltativo) Un FlexMatch matchmaker per creare partite multigiocatore e impostare sessioni di gioco per esse.
- Una o più flotte Amazon GameLift Servers gestite, che utilizzano istanze Amazon Elastic Compute Cloud (Amazon EC2) ottimizzate per l'hosting di giochi multiplayer.

Questa tabella di marcia presenta un percorso semplificato per far funzionare con successo il tuo gioco multiplayer con l'hosting EC2 gestito. Amazon GameLift Servers Dopo aver installato i componenti necessari, puoi continuare a sviluppare giochi e personalizzare la tua soluzione di hosting. Man mano che ti avvicini al lancio, consulta questi articoli [Preparati al lancio con l'Amazon GameLift Servers](#) hosting per aiutarti a preparare la tua soluzione di hosting per l'utilizzo a livello di produzione.

i Inizia subito con il Amazon GameLift Servers plugin per Unreal Engine e Unity

Per una distribuzione più rapida, prova il [Amazon GameLift Servers plug-in](#) per Unreal Engine e Unity. Fornisce flussi di lavoro guidati per l'interfaccia utente per implementare rapidamente il server di gioco con una configurazione minima, in modo da poter provare i componenti del gioco in azione. Quindi puoi basarti su queste basi per creare una soluzione di hosting personalizzata per il tuo gioco. Per ulteriori dettagli, consultare [Prepara il tuo gioco Unreal o Unity con il plugin Amazon GameLift Servers](#).

Passaggio 1: prepara il server di gioco su cui lavorare Amazon GameLift Servers

Aggiungi funzionalità al tuo server di gioco in modo che possa comunicare con il Amazon GameLift Servers servizio quando viene distribuito per l'hosting.

- Scarica l'SDK del server Amazon GameLift Servers (versione 5.x) per il tuo progetto di gioco. L'SDK del server è disponibile in C++, C# e Go. [Scarica un SDK per Amazon GameLift Servers server](#).
- Modifica il codice del server di gioco per aggiungere la funzionalità SDK del server. Per le linee guida, consulta [Prepara un gioco da ospitare con Amazon GameLift Servers](#). Come minimo, procedi come segue:

- Aggiungi codice per inizializzare l'Amazon GameLift Servers SDK e stabilire una WebSocket connessione con il Amazon GameLift Servers servizio. Usa l'azione SDK del server. `InitSdk()`
- Aggiungi codice da segnalare al Amazon GameLift Servers servizio quando il processo del server è pronto per ospitare sessioni di gioco. Usa l'azione `ProcessReady()` SDK del server.
- Implementa le funzioni di callback richieste `OnProcessTerminate()` e `OnStartGameSession()`. Con queste funzioni, i processi del server di gioco possono mantenere una connessione con il Amazon GameLift Servers servizio, avviare una sessione di gioco quando richiesto e rispondere alla richiesta di Amazon GameLift Servers terminare il processo del server di gioco.
- Aggiungi codice per segnalare al Amazon GameLift Servers servizio quando il processo del server sta terminando una sessione di gioco. Usa l'azione `ProcessEnding()` SDK del server.
- Package della build del tuo server di gioco. Crea uno script di installazione con i tuoi file di build, le dipendenze e il software di supporto. Per informazioni, consulta [Creare un pacchetto dei file della build di gioco](#). Ti consigliamo di utilizzare un bucket Amazon Simple Storage Service (Amazon S3) per archiviare le versioni della build del gioco.
- Testa la tua integrazione con i server di gioco. Per questa operazione, consigliamo di configurare una flotta Amazon GameLift Servers Anywhere per una workstation locale, come descritto in [Configura i test locali con Amazon GameLift Servers Anywhere](#). Per questo passaggio, installa manualmente la build del server di gioco sul dispositivo di prova e avvia un processo server. Utilizza la AWS CLI per richiedere una nuova sessione di gioco e verifica che il Amazon GameLift Servers servizio richieda correttamente al processo del server di avviare una sessione di gioco.

Passaggio 2: prepara il client di gioco per partecipare alle sessioni di gioco ospitate

Crea un modo per consentire al client di gioco di richiedere di partecipare a una sessione di gioco, ottenere informazioni di connessione e quindi connettersi direttamente a una sessione di gioco ospitata. L'approccio più comune consiste nel configurare la funzionalità del servizio di backend che funga da intermediario tra il client di gioco e il servizio. Amazon GameLift Servers Questo approccio protegge le tue risorse di hosting e ti offre un maggiore controllo sul modo in cui i giocatori vengono inseriti nelle sessioni di gioco.

- Sviluppa funzionalità di servizio di backend per l'hosting. Il servizio di backend comunica con il Amazon GameLift Servers servizio e fornisce informazioni di connessione a un client di gioco. Questa funzionalità include l'avvio di sessioni di gioco, l'inserimento di giocatori in partite e il recupero delle informazioni sulla sessione di gioco. Per le linee guida, consulta [Prepara un gioco da ospitare con Amazon GameLift Servers](#). Effettua almeno quanto segue:

- Scarica l' AWS SDK per Amazon GameLift Servers e aggiungilo al tuo progetto di servizio di backend. Consulta le [risorse Amazon GameLift Servers SDK per](#) i servizi client.
- Aggiungi codice per inizializzare un Amazon GameLift Servers client e memorizzare le impostazioni chiave. Per informazioni, consulta [Configura l'Amazon GameLift Servers API](#).
- Aggiungi funzionalità per richiamare l'azione AWS SDK `CreateGameSession()` e fornire informazioni sulla connessione della sessione di gioco a un client di gioco. Vedi [Creare una sessione di gioco su una flotta specifica](#).

`CreateGameSession()` La chiamata è un comodo punto di partenza per richiedere nuove sessioni di gioco. Dopo aver implementato un sistema di posizionamento delle sessioni di gioco (vedi Passaggio 3), sostituirai questo codice con una chiamata a `StartGameSessionPlacement()` (o `StartMatchmaking()` se stai utilizzando `FlexMatch`).

Per indicazioni sulla progettazione del servizio di backend, consulta. [Crea un servizio di backend per Amazon GameLift Servers](#)

- Aggiungi funzionalità al tuo client di gioco che consentano ai giocatori di partecipare a una sessione di gioco ospitata. Il client di gioco invia richieste al tuo servizio di backend, non direttamente a Amazon GameLift Servers. Dopo che il servizio di backend ha fornito le informazioni sulla connessione alla sessione di gioco, il client di gioco si connette direttamente alla sessione di gioco per giocare.
- Verifica l'integrazione del tuo client di gioco. Puoi utilizzare la stessa flotta Amazon GameLift Servers Anywhere con una workstation locale per i test.

Per uno sviluppo iterativo rapido o quando lavori con team composti da più persone, ti consigliamo di configurare un ambiente di test [basato sul cloud](#). Questa soluzione Amazon GameLift Servers Toolkit imita il comportamento di una flotta Amazon GameLift Servers gestita ma consente di aggiornare le build dei server di gioco con tempi di risposta minimi.

Passaggio 3: imposta il posizionamento della sessione di gioco

Personalizza il modo in cui desideri Amazon GameLift Servers elaborare le richieste per una nuova sessione di gioco e individua i server di gioco disponibili per ospitarle. Amazon GameLift Servers monitora automaticamente la disponibilità di tutti i server di gioco su tutte le flotte. Quando un client di gioco invia una richiesta di partecipazione a una sessione di gioco, Amazon GameLift Servers cerca il posizionamento «migliore possibile» in base a una serie di priorità definite come latenza minima, costo e disponibilità.

- Crea una coda di sessione di gioco per inserire una nuova sessione di gioco con i server di gioco disponibili. Le code sono il meccanismo principale per il posizionamento delle sessioni di gioco. Per le linee guida, consulta [Crea una coda per le sessioni di gioco](#).
- Come minimo, aggiungi le tue flotte Anywhere come destinazioni nella coda. Tutte le altre impostazioni sono personalizzazioni opzionali.
- Nel codice del servizio di backend, converti la **CreateGameSession()** chiamata in **StartGameSessionPlacement()**. Vedi [Creare una sessione di gioco in una coda con più sedi](#).
- Crea un meccanismo per avvisare un client di gioco quando una sessione di gioco è pronta per partecipare. Durante lo sviluppo, puoi verificare lo stato della sessione di gioco utilizzando una chiamata a DescribeGameSessionPlacement. Prima di utilizzare una coda per elaborare volumi elevati, tuttavia, dovrai abilitare le notifiche degli eventi. Per informazioni, consulta [Imposta la notifica degli eventi per il posizionamento della sessione di gioco](#).
- (Facoltativo) Aggiungi componenti di FlexMatch matchmaking. Per informazioni, consulta la [guida per gli Amazon GameLift ServersFlexMatch sviluppatori](#).

Fase 4: Crea flotte gestite basate sul cloud

Fino a questo punto, hai lavorato con flotte Anywhere autogestite per testare e perfezionare i componenti del gioco e hai ottimizzato il posizionamento delle sessioni di gioco. L'ultima parte della soluzione consiste nel configurare il tipo di risorse di hosting necessarie per un sistema di produzione. Per iniziare a pianificare e configurare la produzione, è necessario passare a lavorare con una flotta Amazon GameLift Servers gestita.

- Package della build del tuo server di gioco e caricala su Amazon GameLift Servers. Crea uno script di installazione con i tuoi file di build, le dipendenze e il software di supporto. Per informazioni, consulta [Crea un server di gioco costruito per Amazon GameLift Servers](#). Puoi caricare la tua build Amazon GameLift Servers utilizzando la console o la AWS CLI.

Prima di caricare la build, decidi in cosa Regione AWS vuoi creare una flotta. Devi caricare la build nella stessa regione. Per ulteriori informazioni sulla scelta dell'ubicazione della flotta, consulta [Posizioni geografiche](#).

- Crea una flotta EC2 gestita. Quando crei una flotta, inizia Amazon GameLift Servers immediatamente a distribuire la versione del tuo server di gioco per l'hosting. Puoi configurare molti aspetti di una flotta gestita. Per le linee guida, consulta [Crea una EC2 flotta Amazon GameLift Servers gestita](#). Come minimo, procedi come segue:
 - Assegna un nome alla flotta e specifica quale build di gioco caricata utilizzare.

- Scegli le istanze On-Demand per il tuo parco istanze e seleziona un tipo di istanza disponibile in base all'ubicazione del tuo parco istanze. Le flotte Spot sono un'opzione preziosa, ma richiedono un design e una configurazione aggiuntivi.
- Crea una configurazione di runtime per la flotta. Specificate almeno il percorso di avvio per il file eseguibile del server di gioco.
- Specificate le impostazioni delle porte per consentire al traffico in entrata di accedere ai server di gioco.
- Aggiungi le flotte gestite alla tua coda. Nella coda delle sessioni di gioco, sostituisci le flotte Anywhere con le flotte gestite.
- Prova l'hosting di giochi con le tue flotte gestite. A questo punto dovresti essere in grado di testare l'intero ciclo di hosting, con un client di gioco che richiede una sessione di gioco, ottiene informazioni di connessione e si connette correttamente a una sessione di gioco.

Fase 5: Personalizza le flotte gestite

Mentre ti prepari per il lancio del gioco, dovrai ottimizzare le tue risorse di hosting gestito. Alcune delle decisioni da prendere in considerazione includono:

- Prendi in considerazione l'aggiunta di flotte Spot per risparmiare sui costi. Per informazioni, consulta [Riduci i costi di hosting dei giochi con le flotte Spot](#).
- Se il tuo server di gioco deve comunicare altre AWS risorse, configura i ruoli IAM per gestire l'accesso. Per informazioni, consulta [Connect il server di gioco Amazon GameLift Servers ospitato ad altre AWS risorse](#).
- Determina dove vuoi posizionare geograficamente i server di gioco. Aggiungi postazioni remote alle tue flotte gestite. Per informazioni, consulta [Personalizzazioni delle risorse di hosting](#).
- Ottimizza le prestazioni del parco macchine selezionando il tipo e la dimensione dell'istanza e configurando il runtime per eseguire più processi server. Per informazioni, consulta [Ottimizza la configurazione del runtime del server di gioco su server gestiti Amazon GameLift Servers](#).
- Sperimenta le opzioni di posizionamento delle sessioni di gioco per le flotte gestite, inclusa la personalizzazione delle impostazioni di prioritizzazione. Per informazioni, consulta [Personalizza una coda di sessioni di gioco](#).
- Imposta il ridimensionamento automatico della capacità per soddisfare la domanda prevista dei giocatori. Per informazioni, consulta [Scalabilità della capacità di hosting di giochi con Amazon GameLift Servers](#).

- Configura flotte di standby in altre Regioni AWS e modifica le code e la scalabilità automatica per gestire i failover, se necessario.
- Configura gli strumenti di osservabilità dell'hosting, tra cui analisi e registrazione. Per informazioni, consulta [Monitoraggio di Amazon GameLift Servers](#).
- Automatizza la distribuzione utilizzando [Infrastructure as Code \(IaC\)](#). Per informazioni, consulta [Gestisci le risorse di Amazon GameLift Servers hosting utilizzando CloudFormation](#).

Amazon GameLift Servers supporta l'uso di AWS CloudFormation modelli per qualsiasi configurazione specifica dell'implementazione. È inoltre possibile utilizzare il per definire le risorse. AWS Cloud Development Kit (AWS CDK) Amazon GameLift Servers Per ulteriori informazioni su AWS CDK, consulta la [Guida per AWS Cloud Development Kit \(AWS CDK\) gli sviluppatori](#).

Per gestire l'implementazione degli CloudFormation stack, consigliamo di utilizzare strumenti e servizi di integrazione continua e distribuzione continua (CI/CD) come. AWS CodePipeline Questi strumenti ti aiutano a implementarlo automaticamente o con l'approvazione ogni volta che crei un file binario per server di gioco. Con CI/CD uno strumento o un servizio, la distribuzione delle risorse per una nuova versione del server di gioco può essere simile a questa:

- Crea e testa il file binario del tuo server di gioco.
- Carica il file binario su Amazon GameLift Servers.
- Implementa nuove flotte con la nuova build.
- Aggiungi le nuove flotte alla coda delle sessioni di gioco e rimuovi le flotte con la versione di build precedente.
- Quando le flotte della build precedente non ospitano più sessioni di gioco attive, elimina le CloudFormation pile di quelle flotte.

Roadmap di sviluppo per l'hosting con contenitori Amazon GameLift Servers gestiti

Questa tabella di marcia ti guida nello sviluppo di una soluzione di hosting Amazon GameLift Servers gestito per i tuoi server di gioco containerizzati. I container gestiti sono solo una soluzione di hosting offerta da. Amazon GameLift Servers Per ulteriori informazioni sulle opzioni di hosting, consulta [Amazon GameLift Servers opzioni di hosting di giochi](#).

Una soluzione container Amazon GameLift Servers gestita con i seguenti componenti:

- Un'immagine del contenitore con la build del server di gioco, caricata nell'archivio privato di Amazon Elastic Container Registry (Amazon ECR). La build del server di gioco è integrata con l'SDK del server per Linux Amazon GameLift Servers e progettata per funzionare su Linux.
- Un servizio di backend che interagisce con il Amazon GameLift Servers servizio per conto dei tuoi client di gioco. Il servizio di backend utilizza le funzionalità dell'API di servizio forAmazon GameLift Servers, che fa parte dell'SDK. AWS
- Una coda di sessioni di Amazon GameLift Servers gioco o un altro meccanismo di posizionamento che elabora le richieste di nuove sessioni di gioco, cerca i server di gioco disponibili in tutte le flotte e richiede a un server di gioco di avviare una sessione di gioco.
- (Facoltativo) Un FlexMatch matchmaker per creare partite multigiocatore e impostare sessioni di gioco per esse.
- Una o più flotte di container, che utilizzano istanze Amazon Elastic Compute Cloud (Amazon EC2) e sono ottimizzate per l'hosting di giochi multiplayer.

Questa tabella di marcia presenta un percorso semplificato per rendere operativi con successo i server di gioco containerizzati con contenitori gestiti. Amazon GameLift Servers Dopo aver installato i componenti necessari, puoi continuare a sviluppare giochi e personalizzare la tua soluzione di hosting. Man mano che ti avvicini al lancio, consulta questi articoli [Preparati al lancio con l'Amazon GameLift Servershosting](#) per aiutarti a preparare la tua soluzione di hosting per l'utilizzo a livello di produzione.

 Accelera l'onboarding con questi strumenti per container gestiti:

- Lo [starter kit per container](#) semplifica l'integrazione e la configurazione della flotta. Aggiunge funzionalità essenziali per la gestione delle sessioni di gioco al server di gioco e utilizza modelli preconfigurati per creare una flotta di container e una pipeline di distribuzione automatizzata per il server di gioco. Dopo l'implementazione, utilizza la Amazon GameLift Servers console e gli strumenti API per monitorare le prestazioni della flotta, gestire le sessioni di gioco e analizzare le metriche.
- Per gli sviluppatori di Unreal Engine e Unity, usa i [Amazon GameLift Serversplugin](#) per integrare il tuo server di gioco e creare una flotta di container dall'interno dell'ambiente di sviluppo del tuo motore di gioco. I flussi di lavoro guidati del plug-in ti aiutano a creare una soluzione semplice e veloce con l'hosting basato su cloud utilizzando contenitori gestiti.

Puoi basarti su queste basi per creare una soluzione di hosting personalizzata per il tuo gioco.

Passaggio 1: prepara il server di gioco su cui lavorare Amazon GameLift Servers

Aggiungi funzionalità al tuo server di gioco in modo che possa comunicare con il Amazon GameLift Servers servizio quando viene distribuito per l'hosting.

- Scarica l'SDK del server Amazon GameLift Servers (versione 5.2 o successiva) per il tuo progetto di gioco. L'SDK del server è disponibile in C++, C# e Go. [Scarica l'SDK del server per Amazon GameLift Servers](#). L'SDK del server è disponibile in C++, C# e Go.
- Modifica il codice del server di gioco per aggiungere la funzionalità SDK del server. Per le linee guida, consulta [Prepara un gioco da ospitare con Amazon GameLift Servers](#). Come minimo, procedi come segue:
 - Aggiungi codice per inizializzare l'Amazon GameLift Servers SDK e stabilire una WebSocket connessione con il Amazon GameLift Servers servizio. Usa l'azione SDK del server. `InitSdk()`
 - Aggiungi codice da segnalare al Amazon GameLift Servers servizio quando il processo del server è pronto per ospitare sessioni di gioco. Usa l'azione `ProcessReady()` SDK del server.
 - Implementa le funzioni di callback richieste `OnStartGameSession()` e `OnProcessTerminate()`. Con queste funzioni, i processi del server di gioco possono mantenere una connessione con il Amazon GameLift Servers servizio, avviare una sessione di gioco quando richiesto e rispondere a una richiesta di Amazon GameLift Servers terminare il processo del server di gioco.
 - Aggiungi codice per segnalare al Amazon GameLift Servers servizio quando il processo del server sta terminando una sessione di gioco. Usa l'azione `ProcessEnding()` SDK del server.
- Package della build del tuo server di gioco. Crea il tuo server di gioco per funzionare su Linux. Prepara la build e gli altri file necessari per eseguire il server di gioco. Se stai sviluppando su Windows, questo passaggio potrebbe comportare la configurazione di un'area di lavoro Linux separata o l'utilizzo di uno strumento come Windows subsystem for Linux (WSL). Avrai bisogno di un ambiente Linux per testare la build del tuo server di gioco e anche per creare e testare le immagini dei tuoi container.
- Metti alla prova l'integrazione del tuo server di gioco. Verifica che il server di gioco integrato sia in grado di connettersi al Amazon GameLift Servers servizio e rispondere alle richieste. Ti consigliamo di configurare una semplice flotta Amazon GameLift Servers Anywhere con una workstation locale come host di test, come descritto in. [Configura i test locali con Amazon GameLift](#)

[Servers Anywhere](#) Installa il tuo server di gioco, costruisci sull'host di test e avvia un processo server. Utilizza la AWS CLI per richiedere una nuova sessione di gioco e verifica che il Amazon GameLift Servers servizio richieda correttamente al processo del server di avviare una sessione di gioco.

Passaggio 2: crea un'immagine del contenitore del server di gioco

Dopo aver integrato con successo il server di gioco, crea un'immagine contenitore con il file eseguibile del server di gioco. Archivalo in un repository Amazon Elastic Container Registry (Amazon ECR) per utilizzarlo con. Amazon GameLift Servers Per istruzioni dettagliate, vedi [Crea un'immagine del contenitore per Amazon GameLift Servers](#).

- Scarica il modello Dockerfile per un contenitore di server di gioco (fornito da). Amazon GameLift Servers Modifica il file per i file di build del tuo server di gioco.
- Crea un'immagine del contenitore del server di gioco. Lavorando in un ambiente Linux, usa lo strumento Docker per creare la tua immagine.
- Invia l'immagine del contenitore ad Amazon ECR. Crea un repository pubblico o privato in Amazon ECR, utilizzandolo Account AWS e Regione AWS dove intendi distribuire la tua flotta di container. Inviaci l'immagine del contenitore.
- Testa le immagini dei container utilizzando la tua flotta Anywhere (opzionale). Potresti voler testare le immagini dei container localmente prima di distribuirle su una flotta di container ospitata nel cloud. Puoi utilizzare la tua flotta Amazon GameLift Servers Anywhere esistente con una workstation locale per i test. Installa ed esegui il contenitore del server di gioco e verifica che: (1) il Amazon GameLift Servers servizio richieda correttamente al processo del server di avviare una sessione di gioco e (2) un client di gioco possa connettersi alla sessione di gioco.

Fase 3: Crea una flotta di container basata sul cloud

Fino a questo punto hai lavorato con una flotta Anywhere autogestita per testare e iterare i componenti del tuo gioco. Se disponi di una versione di server di gioco funzionante e integrata per Amazon GameLift Servers, puoi iniziare a configurare la flotta di container Amazon GameLift Servers gestita basata sul cloud per ospitare le risorse di hosting di cui avrai bisogno per un ambiente di produzione.

- Crea definizioni di gruppi di contenitori. Le definizioni dei gruppi di contenitori descrivono l'architettura dei container per una flotta e identificano quali immagini di container distribuire. Per informazioni, consulta [Creare una definizione di gruppo di container per una flotta di Amazon](#)

[GameLift Servers container](#). Crea la definizione del gruppo di contenitori nello stesso Regione AWS luogo in cui sono archiviate le immagini dei contenitori. Per ulteriori informazioni sulla scelta dell'ubicazione della flotta, consulta [Posizioni geografiche](#). Come minimo, procedi come segue:

- Crea una definizione del gruppo di contenitori del server di gioco.
- Aggiungi una definizione di contenitore con un'immagine del contenitore con la build del tuo server di gioco.
- Configura un intervallo di porte per i processi del server di gioco del contenitore.
- Crea una flotta di container gestita. Quando crei una flotta, inizia Amazon GameLift Servers immediatamente a distribuire il server di gioco creato per l'hosting. Puoi configurare molti aspetti di una flotta gestita. Per le linee guida, consulta [Crea una flotta di container Amazon GameLift Servers gestita](#). Come minimo, procedi come segue:
 - Imposta un ruolo di servizio AWS Identity and Access Management (IAM) per la flotta di container. Per informazioni, consulta [Configurare un ruolo di servizio IAM per Amazon GameLift Servers](#).
 - Specificate la definizione del gruppo di contenitori del server di gioco da distribuire sulle istanze della flotta.
 - Usa i valori predefiniti, se disponibili, per tutti gli altri parametri. Amazon GameLift Servers calcola alcuni parametri per una configurazione ottimale.
- Aggiungi le flotte di container alla tua coda. Nella coda della sessione di gioco, sostituisci la flotta di test Anywhere con la tua flotta di container gestita.
- Metti alla prova l'hosting di giochi con le tue flotte di container. A questo punto dovresti essere in grado di testare l'intera soluzione. Avvia un client di gioco e richiedi una sessione di gioco tramite il servizio di backend. Ottieni informazioni sulla connessione e connettiti a una sessione di gioco sulla flotta di container.
- Ripeti le implementazioni della tua flotta. Puoi aggiornare le definizioni dei gruppi di container e le configurazioni della flotta e quindi distribuire aggiornamenti alle flotte esistenti.

Passaggio 4: Prepara il client di gioco per partecipare alle sessioni di gioco ospitate

Crea un modo per consentire al client di gioco di richiedere di partecipare a una sessione di gioco, ottenere informazioni di connessione e quindi connettersi direttamente a una sessione di gioco ospitata. L'approccio più comune consiste nel configurare la funzionalità del servizio di backend che funga da intermediario tra il client di gioco e il servizio. Amazon GameLift Servers Questo approccio protegge le tue risorse di hosting e ti offre un maggiore controllo sul modo in cui i giocatori vengono inseriti nelle sessioni di gioco.

- Sviluppa funzionalità di servizio di backend per l'hosting. Il servizio di backend comunica con il Amazon GameLift Servers servizio e fornisce informazioni di connessione a un client di gioco. Questa funzionalità include l'avvio di sessioni di gioco, l'inserimento di giocatori in partite e il recupero delle informazioni sulla sessione di gioco. Per le linee guida, consulta [Prepara un gioco da ospitare con Amazon GameLift Servers](#). Effettua almeno quanto segue:
- Scarica l' AWS SDK per Amazon GameLift Servers e aggiungilo al tuo progetto di servizio di backend. Consulta le [risorse Amazon GameLift Servers SDK per](#) i servizi client.
- Aggiungi codice per inizializzare un Amazon GameLift Servers client e memorizzare le impostazioni chiave. Per informazioni, consulta [Configura l'Amazon GameLift Servers API](#).
- Aggiungi funzionalità per richiamare l'azione AWS SDK `CreateGameSession()` e fornire informazioni sulla connessione della sessione di gioco a un client di gioco. Vedi [Creare una sessione di gioco su una flotta specifica](#).

`CreateGameSession()` La chiamata è un comodo punto di partenza per richiedere nuove sessioni di gioco. Dopo aver implementato un sistema di posizionamento delle sessioni di gioco (vedi Passaggio 3), sostituirai questo codice con una chiamata a `StartGameSessionPlacement()` (o `StartMatchmaking()` se stai utilizzando `FlexMatch`).

Per indicazioni sulla progettazione del servizio di backend, consulta. [Crea un servizio di backend per Amazon GameLift Servers](#)

- Aggiungi funzionalità al tuo client di gioco che consentano ai giocatori di partecipare a una sessione di gioco ospitata. Il client di gioco invia richieste al tuo servizio di backend, non direttamente a Amazon GameLift Servers. Dopo che il servizio di backend ha fornito le informazioni sulla connessione alla sessione di gioco, il client di gioco si connette direttamente alla sessione di gioco per giocare.
- Verifica l'integrazione del tuo client di gioco. Puoi utilizzare la tua flotta Amazon GameLift Servers Anywhere esistente con una workstation locale per i test. Utilizza il nuovo servizio di backend per richiedere una nuova sessione di gioco e verifica che: (1) il Amazon GameLift Servers servizio richieda correttamente al processo del server di avviare una sessione di gioco e (2) un client di gioco possa connettersi alla sessione di gioco.

Passaggio 5: Imposta il posizionamento della sessione di gioco

Personalizza il modo in cui desideri Amazon GameLift Servers elaborare le richieste per una nuova sessione di gioco e individua i server di gioco disponibili per ospitarle. Amazon GameLift Servers monitora automaticamente la disponibilità di tutti i server di gioco su tutte le flotte. Quando

un client di gioco invia una richiesta di partecipazione a una sessione di gioco, Amazon GameLift Servers cerca il posizionamento «migliore possibile» in base a una serie di priorità definite come latenza minima, costo e disponibilità.

- Crea una coda di sessione di gioco per inserire una nuova sessione di gioco con i server di gioco disponibili. Le code sono il meccanismo principale per il posizionamento delle sessioni di gioco. Per le linee guida, consulta [Crea una coda per le sessioni di gioco](#).
- Come minimo, aggiungi le tue flotte Anywhere come destinazioni nella coda. Tutte le altre impostazioni sono personalizzazioni opzionali.
- Nel codice del servizio di backend, converti la **CreateGameSession()** chiamata in **StartGameSessionPlacement()**. Vedi [Creare una sessione di gioco in una coda con più sedi](#).
- Crea un meccanismo per avvisare un client di gioco quando una sessione di gioco è pronta per partecipare. Durante lo sviluppo, puoi verificare lo stato della sessione di gioco utilizzando una chiamata a `DescribeGameSessionPlacement`. Prima di utilizzare una coda per elaborare volumi elevati, tuttavia, dovrai abilitare le notifiche degli eventi. Per informazioni, consulta [Imposta la notifica degli eventi per il posizionamento della sessione di gioco](#).
- Aggiungi FlexMatch matchmaking (opzionale). Crea un set di regole di matchmaking e crea una configurazione di matchmaking per funzionare con la coda della tua sessione di gioco. [Per indicazioni sulla configurazione di un sistema di matchmaking, consulta la guida per gli sviluppatori. Amazon GameLift Servers FlexMatch](#)
- Prova il sistema di posizionamento. Puoi utilizzare la tua flotta Amazon GameLift Servers Anywhere esistente con una workstation locale per i test. Utilizza il servizio di backend per richiedere una nuova sessione di gioco e verifica che il Amazon GameLift Servers servizio richieda correttamente al processo del server di avviare una sessione di gioco.

Fase 6: Personalizza la tua soluzione container gestita

Mentre ti prepari per il lancio del gioco, dovrai ottimizzare le tue risorse di hosting gestito. Alcune delle decisioni da prendere in considerazione includono:

- Ottimizza la configurazione della tua flotta di container. Per informazioni, consulta [Personalizza una flotta di Amazon GameLift Servers container](#).
- Prendi in considerazione l'aggiunta di flotte Spot per risparmiare sui costi. Per informazioni, consulta [Riduci i costi di hosting dei giochi con le flotte Spot](#).

- Se il tuo server di gioco deve comunicare altre AWS risorse, configura i ruoli IAM per gestire l'accesso. Per informazioni, consulta [Connect il server di gioco Amazon GameLift Servers ospitato ad altre AWS risorse](#).
- Determina dove vuoi posizionare geograficamente i server di gioco. Aggiungi postazioni remote alle tue flotte gestite. Per informazioni, consulta [Personalizzazioni delle risorse di hosting](#).
- Sperimenta le opzioni di posizionamento delle sessioni di gioco per le flotte gestite, inclusa la personalizzazione delle impostazioni di prioritizzazione. Per informazioni, consulta [Personalizza una coda di sessioni di gioco](#).
- Imposta il ridimensionamento automatico della capacità per soddisfare la domanda prevista dei giocatori. Per informazioni, consulta [Scalabilità della capacità di hosting di giochi con Amazon GameLift Servers](#).
- Crea flotte in altre Regioni AWS e modifica le code e la scalabilità automatica per gestire i failover secondo necessità.
- Configura gli strumenti di osservabilità dell'hosting, tra cui analisi e registrazione. Per informazioni, consulta [Monitoraggio di Amazon GameLift Servers](#).
- Automatizza le implementazioni della tua flotta utilizzando l'[infrastruttura come](#) codice (IaC). Per informazioni, consulta [Gestisci le risorse di Amazon GameLift Servers hosting utilizzando CloudFormation](#).

Amazon GameLift Servers supporta l'uso di AWS CloudFormation modelli per qualsiasi configurazione specifica dell'implementazione. È inoltre possibile utilizzare il per definire le risorse. AWS Cloud Development Kit (AWS CDK) Amazon GameLift Servers Per ulteriori informazioni su AWS CDK, consulta la [Guida per AWS Cloud Development Kit \(AWS CDK\) gli sviluppatori](#).

Per gestire l'implementazione degli CloudFormation stack, consigliamo di utilizzare strumenti e servizi di integrazione continua e distribuzione continua (CI/CD) come. AWS CodePipeline Questi strumenti ti aiutano a implementarlo automaticamente o con l'approvazione ogni volta che crei un file binario per server di gioco. Con CI/CD uno strumento o un servizio, la distribuzione delle risorse per una nuova versione del server di gioco può essere simile a questa:

- Crea e testa il file binario del tuo server di gioco.
- Carica il file binario su Amazon GameLift Servers.
- Implementa nuove flotte con la nuova build.
- Aggiungi le nuove flotte alla coda delle sessioni di gioco e rimuovi le flotte con la versione di build precedente.

- Quando le flotte della build precedente non ospitano più sessioni di gioco attive, elimina le CloudFormation pile di quelle flotte.

Roadmap di sviluppo per l'hosting con Anywhere Amazon GameLift Servers

Questa tabella di marcia illustra come sviluppare una soluzione di hosting per il gioco multiplayer da utilizzare con le tue risorse (hardware locale o macchine virtuali). Amazon GameLift Servers offre diverse opzioni di hosting di giochi; per ulteriori informazioni su queste opzioni, consulta [Amazon GameLift Servers opzioni di hosting di giochi](#)

Con Amazon GameLift Servers Anywhere hosting, il tuo server di gioco è ospitato su risorse informatiche che fornisci e gestisci. Puoi creare una flotta Anywhere con le configurazioni di cui hai bisogno e dislocata geograficamente ovunque si trovino i tuoi giocatori. Amazon GameLift Servers offre le seguenti funzionalità per una flotta Anywhere:

- Gestisce per te il processo di posizionamento delle sessioni di gioco in base alla tua configurazione, tra cui:
 - Monitoraggio della disponibilità dei server di gioco nelle tue flotte Anywhere.
 - Elaborazione delle richieste di gioco dal servizio client di gioco e abbinamento delle richieste di gioco ai server disponibili.
 - Richiedere ai server di gioco delle flotte Anywhere di avviare sessioni di gioco.
 - Comunicazione dei dettagli di connessione ai client di gioco.
- Raccoglie le metriche delle prestazioni per il processo di posizionamento delle sessioni e le metriche di utilizzo per le sessioni di gioco e i giocatori.
- Supporta il set completo di funzionalità di FlexMatch matchmaking, in modo da poter creare un matchmaker e integrarlo con il sistema di posizionamento delle sessioni di gioco.
- Offre all'Amazon GameLift Servers agente la gestione delle principali attività di gestione degli host su una flotta Anywhere.
- Supporta la combinazione con flotte Amazon GameLift Servers gestite per una soluzione ibrida flessibile.

Una soluzione Amazon GameLift Servers Anywhere è composta dai seguenti componenti:

- Una versione di server di gioco, integrata con l'SDK del server per Amazon GameLift Servers, da distribuire su tutte le flotte.

- Un client di gioco e un servizio di backend, integrato con l' AWS SDK, per interagire con il Amazon GameLift Servers servizio e richiedere sessioni di gioco.
- Una Amazon GameLift Servers coda o un altro meccanismo di posizionamento per inserire nuove sessioni di gioco con i server di gioco disponibili in tutte le flotte.
- (Facoltativo) Un FlexMatch matchmaker per creare partite multigiocatore e impostare sessioni di gioco per esse.
- Una o più flotte Amazon GameLift Servers Anywhere con risorse di hosting locali o di altro tipo, gestite con gli strumenti di configurazione, gestione e distribuzione esistenti. (È possibile utilizzare facoltativamente il.) AWS Systems Manager

Questa tabella di marcia presenta un percorso semplificato per avviare e far funzionare con successo il gioco multiplayer con Amazon GameLift Servers l'hosting Anywhere. Dopo aver installato i componenti necessari, puoi continuare a sviluppare giochi e personalizzare la tua soluzione di hosting. Man mano che ti avvicini al lancio, consulta questi articoli [Preparati al lancio con l'Amazon GameLift Servers hosting](#) per aiutarti a preparare la tua soluzione di hosting per l'utilizzo a livello di produzione.

i Inizia subito con il Amazon GameLift Servers plugin per Unreal Engine e Unity

Per una distribuzione più rapida, prova il [Amazon GameLift Servers plugin](#) per Unreal Engine e Unity. Fornisce flussi di lavoro guidati per l'interfaccia utente per implementare rapidamente il server di gioco con una configurazione minima, in modo da poter provare i componenti del gioco in azione. Quindi puoi basarti su queste basi per creare una soluzione di hosting personalizzata per il tuo gioco. Per ulteriori dettagli, consultare [Prepara il tuo gioco Unreal o Unity con il plugin Amazon GameLift Servers](#).

Passaggio 1: prepara il server di gioco su cui lavorare Amazon GameLift Servers

Aggiungi funzionalità al tuo server di gioco in modo che possa comunicare con il Amazon GameLift Servers servizio quando viene distribuito per l'hosting.

- Scarica l'SDK del server Amazon GameLift Servers (versione 5.x) per il tuo progetto di gioco. L'SDK del server è disponibile in C++, C# e Go. [Scarica un SDK per Amazon GameLift Servers server](#).

- Modifica il codice del server di gioco per aggiungere la funzionalità SDK del server. Per le linee guida, consulta [Prepara un gioco da ospitare con Amazon GameLift Servers](#). Come minimo, procedi come segue:
 - Aggiungi codice per inizializzare l'Amazon GameLift Servers SDK e stabilire una WebSocket connessione con il Amazon GameLift Servers servizio. Utilizza l'azione SDK del server `InitSdk()` e includi i parametri del server, necessari per una flotta Anywhere.
 - Aggiungi codice per segnalare al Amazon GameLift Servers servizio quando il processo del server è pronto per ospitare sessioni di gioco. Usa l'azione `ProcessReady()` SDK del server.
 - Implementa le funzioni di callback richieste `OnProcessTerminate()` e `OnStartGameSession()`. Con queste funzioni, i processi del server di gioco possono mantenere una connessione con il Amazon GameLift Servers servizio, avviare una sessione di gioco quando richiesto e rispondere alla richiesta di Amazon GameLift Servers terminare il processo del server di gioco.
 - Aggiungi codice per segnalare al Amazon GameLift Servers servizio quando il processo del server sta terminando una sessione di gioco. Usa l'azione `ProcessEnding()` SDK del server.
- Package della build del tuo server di gioco. Crea uno script di installazione con i tuoi file di build, le dipendenze e il software di supporto. Per informazioni, consulta [Creare un pacchetto dei file della build di gioco](#). Ti consigliamo di utilizzare un bucket Amazon Simple Storage Service (Amazon S3) per archiviare le versioni della build del gioco.
- Testa la tua integrazione con i server di gioco. Per questa operazione, consigliamo di configurare una flotta Amazon GameLift Servers Anywhere per una workstation locale, come descritto in [Configura i test locali con Amazon GameLift Servers Anywhere](#). Per questo passaggio, installa manualmente la build del server di gioco sul dispositivo di prova e avvia un processo server. Utilizza la AWS CLI per richiedere una nuova sessione di gioco e verifica che il Amazon GameLift Servers servizio richieda correttamente al processo del server di avviare una sessione di gioco.

Passaggio 2: prepara il client di gioco per partecipare alle sessioni di gioco ospitate

Crea un modo per consentire al client di gioco di richiedere di partecipare a una sessione di gioco, ottenere informazioni di connessione e quindi connettersi direttamente a una sessione di gioco ospitata. L'approccio più comune consiste nel configurare la funzionalità del servizio di backend che funga da intermediario tra il client di gioco e il servizio. Amazon GameLift Servers Questo approccio protegge le tue risorse di hosting e ti offre un maggiore controllo sul modo in cui i giocatori vengono inseriti nelle sessioni di gioco.

- Sviluppa funzionalità di servizio di backend per l'hosting. Il servizio di backend comunica con il Amazon GameLift Servers servizio e fornisce informazioni di connessione a un client di gioco. Questa funzionalità include l'avvio di sessioni di gioco, l'inserimento di giocatori in partite e il recupero delle informazioni sulla sessione di gioco. Per le linee guida, consulta [Prepara un gioco da ospitare con Amazon GameLift Servers](#). Effettua almeno quanto segue:
- Scarica l' AWS SDK per Amazon GameLift Servers e aggiungilo al tuo progetto di servizio di backend. Consulta le [risorse Amazon GameLift Servers SDK per](#) i servizi client.
- Aggiungi codice per inizializzare un Amazon GameLift Servers client e memorizzare le impostazioni chiave. Per informazioni, consulta [Configura l'Amazon GameLift Servers API](#).
- Aggiungi funzionalità per richiamare l'azione AWS SDK `CreateGameSession()` e fornire informazioni sulla connessione della sessione di gioco a un client di gioco. Vedi [Creare una sessione di gioco su una flotta specifica](#).

`CreateGameSession()` La chiamata è un comodo punto di partenza per richiedere nuove sessioni di gioco. Dopo aver implementato un sistema di posizionamento delle sessioni di gioco (vedi Passaggio 3), sostituirai questo codice con una chiamata a `StartGameSessionPlacement()` (o `StartMatchmaking()` se stai utilizzando `FlexMatch`).

Per indicazioni sulla progettazione del servizio di backend, consulta. [Crea un servizio di backend per Amazon GameLift Servers](#)

- Aggiungi funzionalità al tuo client di gioco che consentano ai giocatori di partecipare a una sessione di gioco ospitata. Il client di gioco invia richieste al tuo servizio di backend, non direttamente a Amazon GameLift Servers. Dopo che il servizio di backend ha fornito le informazioni sulla connessione alla sessione di gioco, il client di gioco si connette direttamente alla sessione di gioco per giocare.
- Verifica l'integrazione del tuo client di gioco. Puoi utilizzare la stessa flotta Amazon GameLift Servers Anywhere con workstation locali per i test.

Passaggio 3: imposta il posizionamento della sessione di gioco

Personalizza il modo in cui desideri Amazon GameLift Servers elaborare le richieste per una nuova sessione di gioco e individua i server di gioco disponibili per ospitarle. Amazon GameLift Servers monitora automaticamente la disponibilità di tutti i server di gioco su tutte le flotte. Quando un client di gioco invia una richiesta di partecipazione a una sessione di gioco, Amazon GameLift Servers cerca il posizionamento «migliore possibile» in base a una serie di priorità definite come latenza minima, costo e disponibilità.

- Crea una coda di sessione di gioco per inserire una nuova sessione di gioco con i server di gioco disponibili. Le code sono il meccanismo principale per il posizionamento delle sessioni di gioco. Per le linee guida, consulta [Crea una coda per le sessioni di gioco](#).
- Come minimo, aggiungi le tue flotte Anywhere come destinazioni nella coda. Tutte le altre impostazioni sono facoltative.
- Nel codice del servizio di backend, converti la **CreateGameSession()** chiamata in **StartGameSessionPlacement()** Vedi [Creare una sessione di gioco in una coda con più sedi](#).
- Crea un meccanismo per avvisare un client di gioco quando una sessione di gioco è pronta per partecipare. Durante lo sviluppo, puoi verificare lo stato della sessione di gioco utilizzando una chiamata a DescribeGameSessionPlacement. Prima di utilizzare una coda per elaborare volumi elevati, tuttavia, dovrai abilitare le notifiche degli eventi. Per informazioni, consulta [Imposta la notifica degli eventi per il posizionamento della sessione di gioco](#).
- (Facoltativo) Aggiungi componenti di FlexMatch matchmaking. Per informazioni, consulta la [guida per gli Amazon GameLift ServersFlexMatch sviluppatori](#).

Fase 4: Configura una flotta Anywhere con l'Amazon GameLift Serversagente

Fino a questo punto hai utilizzato dispositivi locali (registrati come Anywhere fleet computes) per testare e iterare i componenti del gioco. Il passaggio successivo consiste nel configurare il tipo di flotta di cui avrai bisogno per un sistema di produzione. Per queste risorse, utilizza l'Amazon GameLift Serversagente per gestire alcune attività chiave di gestione degli host in ambito informatico. Per ulteriori dettagli, consultare [Collabora con l'Amazon GameLift Serversagente](#).

- Scarica l'Amazon GameLift Serversagente e aggiungilo al pacchetto di installazione del server di gioco. Ottieni e crea il codice sorgente dell'agente, disponibile nel [GitHub repository dell'Amazon GameLift Serversagente](#). Posiziona il file JAR risultante eseguibile nella stessa directory dell'eseguibile della build del gioco.
- Se necessario, modificate lo script di avvio per l'agente. Assicurati che l'eseguibile dell'agente venga avviato non appena un computer inizia a funzionare. Consulta il file readme nel repository Agent per informazioni sull'installazione e l'esecuzione dell'agente sui computer di hosting. Il comando di avvio dovrebbe includere opzioni per specificare, come minimo, l'Anywhere fleet ID e Regione AWS una posizione personalizzata e un nome di calcolo.

L'agente gestisce automaticamente le seguenti attività per te, quindi se hai gestito queste attività con script, puoi rimuoverle:

- Chiamate RegisterCompute() per aggiungere l'elaborazione a una flotta Anywhere.

- Chiamate `GetComputeAuthToken()` per autenticare i server di gioco quando si connettono al Amazon GameLift Servers servizio. L'agente gestisce il recupero e l'aggiornamento del token di autenticazione, che può essere utilizzato da tutti i processi del server di gioco in esecuzione sul computer.
- Avvia nuovi processi del server sul computer in base a una serie di istruzioni di runtime.
- Crea una configurazione di runtime per i computer della tua flotta Anywhere. Puoi utilizzare la Amazon GameLift Servers console o la AWS CLI per creare o modificare le istruzioni di runtime per la flotta. L'agente esegue queste istruzioni e richiede periodicamente aggiornamenti al Amazon GameLift Servers servizio.
- Configura o modifica la coda delle sessioni di gioco in base alle tue esigenze. Crea una nuova coda (o aggiornane una esistente) per utilizzare le flotte Anywhere distribuite con l'agente. Amazon GameLift Servers
- Testa l'integrazione di Agent con le tue flotte Anywhere. Verifica che l'agente stia avviando correttamente i processi del server in base alla configurazione di runtime.

Fase 5: Personalizza le tue flotte Anywhere

Mentre ti prepari per il lancio del gioco, dovrai ottimizzare le tue risorse di hosting gestito. Alcune delle decisioni da prendere in considerazione includono:

- Automatizza il processo di avvio e spegnimento dei computer in base alle esigenze, inclusa l'installazione e l'esecuzione del software del server di gioco. Il riciclaggio dei computer è utile per garantire che vengano aggiornati regolarmente e spegnerli può far risparmiare sui costi quando non sono necessari.
- Se il tuo server di gioco deve comunicare altre AWS risorse, configura i ruoli IAM per gestire l'accesso. Per informazioni, consulta [Connect il server di gioco Amazon GameLift Servers ospitato ad altre AWS risorse](#).
- Determina dove vuoi posizionare geograficamente i server di gioco. Aggiungi postazioni remote alle tue flotte gestite. Per informazioni, consulta [Personalizzazioni delle risorse di hosting](#).
- Ottimizza le prestazioni del parco macchine selezionando le configurazioni delle risorse di calcolo, quindi configura le istruzioni di runtime per eseguire un numero ottimale di processi server per elaborazione.
- Sperimenta le opzioni di posizionamento delle sessioni di gioco per flotte gestite, inclusa la personalizzazione delle impostazioni di prioritizzazione. Per informazioni, consulta [Personalizza una coda di sessioni di gioco](#).

- Crea meccanismi per gestire la scalabilità manuale o automatica della capacità per soddisfare la domanda prevista dei giocatori. Considerate quali fattori dovrebbero indurre il sistema ad aumentare o diminuire il numero di computer disponibili per ospitare sessioni di gioco.
- Progetta e implementa il failover su altre risorse, se necessario.
- Configura strumenti di osservabilità dell'hosting, tra cui analisi e registrazione. Per informazioni, consulta [Monitoraggio di Amazon GameLift Servers](#).

Roadmap di sviluppo per l'hosting ibrido con Amazon GameLift Servers

Questa tabella di marcia ti guida nello sviluppo di una soluzione di hosting per il tuo gioco multiplayer. Amazon GameLift Servers offre diverse opzioni di hosting di giochi; per ulteriori informazioni su queste opzioni, consulta [Amazon GameLift Servers opzioni di hosting di giochi](#).

Una soluzione ibrida utilizza una combinazione di risorse di hosting, tra cui risorse basate sul cloud gestite da Amazon GameLift Servers e risorse di hosting gestite autonomamente dall'utente. Per una discussione più dettagliata sull'hosting ibrido, consulta questo articolo: [Hosting di server di gioco ibridi](#) con Anywhere. Con Amazon GameLift Servers, puoi configurare una soluzione ibrida che utilizza componenti e processi comuni, in modo da gestire centralmente una flotta globale e spostare facilmente i carichi tra tutti i tipi di risorse.

Un'architettura ibrida è composta dai seguenti componenti:

- Una singola build di server di gioco, integrata con l'SDK del server per Amazon GameLift Servers, da distribuire su tutte le flotte.
- Un unico client di gioco e servizio di backend, integrato con l'AWS SDK, per interagire con il Amazon GameLift Servers servizio e richiedere sessioni di gioco.
- Una Amazon GameLift Servers coda condivisa per inserire nuove sessioni di gioco con i server di gioco disponibili e bilanciare il carico su tutte le flotte.
- (Facoltativo) Un agente di gestione dei processi, che viene distribuito con una flotta Anywhere, per semplificare le attività di gestione dei processi del server su tutti i computer di tutte le flotte.
- (Facoltativo) Un FlexMatch matchmaker per creare partite multigiocatore e impostare sessioni di gioco per esse.
- Una o più flotte Amazon GameLift Servers gestite, che utilizzano istanze Amazon Elastic Compute Cloud (Amazon EC2) ottimizzate per l'hosting di giochi multiplayer.

- Una o più flotte Amazon GameLift Servers Anywhere, che utilizzano le tue risorse locali o di altro tipo di hosting esistenti, inclusi gli strumenti di configurazione, gestione e distribuzione. (Facoltativamente, puoi usare il.) AWS Systems Manager

Questa tabella di marcia presenta un percorso semplificato per avviare e far funzionare con successo il gioco multiplayer in una soluzione di hosting ibrida con Amazon GameLift Servers. Dopo aver installato i componenti necessari, puoi continuare a sviluppare giochi e personalizzare la tua soluzione di hosting. Man mano che ti avvicini al lancio, consulta questi articoli [Preparati al lancio con l'Amazon GameLift Servers hosting](#) per aiutarti a preparare la tua soluzione di hosting per l'utilizzo a livello di produzione.

Inizia subito con il plugin Amazon GameLift Servers

Se stai sviluppando progetti con Unreal Engine o Unity, inizia a configurare il gioco per l'hosting con il Amazon GameLift Servers plug-in. Con il plug-in, puoi aggiungerlo Amazon GameLift Servers SDKs al tuo progetto di gioco e utilizzare i flussi di lavoro guidati per creare una versione semplice e funzionante di una soluzione di hosting ibrida con una flotta Anywhere e una flotta Amazon GameLift Servers gestita. Puoi quindi utilizzare questi principi fondamentali per svilupparli e personalizzarli secondo necessità.

Passaggio 1: prepara il server di gioco su cui lavorare Amazon GameLift Servers

Aggiungi funzionalità al tuo server di gioco in modo che possa comunicare con il Amazon GameLift Servers servizio quando viene distribuito per l'hosting. La stessa funzionalità è richiesta se il server di gioco è in esecuzione su una flotta Amazon GameLift Servers gestita o su una flotta Anywhere.

- Scarica l'SDK del server Amazon GameLift Servers (versione 5.x) per il tuo progetto di gioco. L'SDK del server è disponibile in C++, C# e Go. [Scarica l'SDK del server per Amazon GameLift Servers](#)
- Modifica il codice del server di gioco per aggiungere la funzionalità SDK del server. Per le linee guida, consulta [Prepara un gioco da ospitare con Amazon GameLift Servers](#). Come minimo, procedi come segue:
 - Aggiungi codice per inizializzare l'Amazon GameLift Servers SDK e stabilire una WebSocket connessione con il Amazon GameLift Servers servizio. Usa l'azione SDK del server. `InitSdk()` Includi il codice per specificare i parametri del server durante l'esecuzione su un computer Anywhere fleet.

- Aggiungi codice per segnalare al Amazon GameLift Servers servizio quando il processo del server è pronto per ospitare sessioni di gioco. Usa l'azione `ProcessReady()` SDK del server.
- Implementa le funzioni di callback richieste `OnProcessTerminate()` e `OnStartGameSession()`. Con queste funzioni, i processi del server di gioco possono mantenere una connessione con il Amazon GameLift Servers servizio, avviare una sessione di gioco quando richiesto e rispondere alla richiesta di Amazon GameLift Servers terminare il processo del server di gioco.
- Aggiungi codice per segnalare al Amazon GameLift Servers servizio quando il processo del server sta terminando una sessione di gioco. Usa l'azione `ProcessEnding()` SDK del server.
- Package della build del tuo server di gioco. Crea uno script di installazione con i tuoi file di build, le dipendenze e il software di supporto. Per informazioni, consulta [Creare un pacchetto dei file della build di gioco](#). Ti consigliamo di utilizzare un bucket Amazon Simple Storage Service (Amazon S3) per archiviare le versioni della build del gioco.
- Testa la tua integrazione con i server di gioco. Per questa operazione, consigliamo di configurare una flotta Amazon GameLift Servers Anywhere con una workstation locale, come descritto in [Configura i test locali con Amazon GameLift Servers Anywhere](#). Per questo passaggio, installa manualmente la build del server di gioco sul dispositivo di prova e avvia un processo server. Utilizza la AWS CLI per richiedere una nuova sessione di gioco e verifica che il Amazon GameLift Servers servizio richieda correttamente al processo del server di avviare una sessione di gioco.

Passaggio 2: prepara il client di gioco per partecipare alle sessioni di gioco ospitate

Crea un modo per consentire al client di gioco di richiedere di partecipare a una sessione di gioco, ottenere informazioni di connessione e quindi connettersi direttamente a una sessione di gioco ospitata. L'approccio più comune consiste nel configurare la funzionalità del servizio di backend che funga da intermediario tra il client di gioco e il servizio. Amazon GameLift Servers Questo approccio protegge le tue risorse di hosting e ti offre un maggiore controllo sul modo in cui i giocatori vengono inseriti nelle sessioni di gioco.

- Sviluppa funzionalità di servizio di backend per l'hosting. Il servizio di backend comunica con il Amazon GameLift Servers servizio e fornisce informazioni di connessione a un client di gioco. Questa funzionalità include l'avvio di sessioni di gioco, l'inserimento di giocatori in partite e il recupero delle informazioni sulla sessione di gioco. Per le linee guida, consulta [Prepara un gioco da ospitare con Amazon GameLift Servers](#). Effettua almeno quanto segue:
- Scarica l' AWS SDK per Amazon GameLift Servers e aggiungilo al tuo progetto di servizio di backend. Consulta le [risorse Amazon GameLift Servers SDK per i servizi client](#).

- Aggiungi codice per inizializzare un Amazon GameLift Servers client e memorizzare le impostazioni chiave. Per informazioni, consulta [Configura l'Amazon GameLift Servers API](#).
- Aggiungi funzionalità per richiamare l'azione dell' AWS SDK `CreateGameSession()` e fornire informazioni sulla connessione della sessione di gioco a un client di gioco. Vedi [Creare una sessione di gioco su una flotta specifica](#).

`CreateGameSession()` La chiamata è un comodo punto di partenza per richiedere nuove sessioni di gioco. Dopo aver implementato un sistema di posizionamento delle sessioni di gioco (vedi Passaggio 3), sostituirai questo codice con una chiamata a `StartGameSessionPlacement()` (o `StartMatchmaking()` se stai utilizzando `FlexMatch`).

Per indicazioni sulla progettazione del servizio di backend, consulta. [Crea un servizio di backend per Amazon GameLift Servers](#)

- Aggiungi funzionalità al tuo client di gioco che consentano ai giocatori di partecipare a una sessione di gioco ospitata. Il client di gioco invia richieste al tuo servizio di backend, non direttamente a Amazon GameLift Servers. Dopo che il servizio di backend ha fornito le informazioni sulla connessione alla sessione di gioco, il client di gioco si connette direttamente alla sessione di gioco per giocare.
- Verifica l'integrazione del tuo client di gioco. Puoi utilizzare la stessa flotta Amazon GameLift Servers Anywhere con una workstation locale per i test.

Durante la fase di sviluppo, se desideri testare il comportamento della build del tuo gioco in una flotta Amazon GameLift Servers gestita, ti consigliamo di configurare anche un ambiente di test [basato sul cloud](#). Questa soluzione Amazon GameLift Servers Toolkit imita il comportamento di una flotta gestita, ma consente di aggiornare le build dei server di gioco con tempi di risposta minimi.

Passaggio 3: imposta il posizionamento della sessione di gioco

Personalizza il modo in cui desideri Amazon GameLift Servers elaborare le richieste per una nuova sessione di gioco e individua i server di gioco disponibili per ospitarle. Amazon GameLift Servers monitora automaticamente la disponibilità di tutti i server di gioco su tutte le flotte. Quando un client di gioco invia una richiesta di partecipazione a una sessione di gioco, Amazon GameLift Servers cerca il posizionamento «migliore possibile» in base a una serie di priorità definite come latenza minima, costo e disponibilità.

- Crea una coda di sessione di gioco per inserire una nuova sessione di gioco con i server di gioco disponibili. Le code sono il meccanismo principale per il posizionamento delle sessioni di gioco. Per le linee guida, consulta [Crea una coda per le sessioni di gioco](#).
- Come minimo, aggiungi le tue flotte Anywhere come destinazioni nella coda. Tutte le altre impostazioni sono personalizzazioni opzionali.
- Nel codice del servizio di backend, converti la **CreateGameSession()** chiamata in **StartGameSessionPlacement()** Vedi [Creare una sessione di gioco in una coda con più sedi](#).
- Crea un meccanismo per avvisare un client di gioco quando una sessione di gioco è pronta per partecipare. Durante lo sviluppo, puoi verificare lo stato della sessione di gioco utilizzando una chiamata a DescribeGameSessionPlacement. Prima di utilizzare una coda per elaborare volumi elevati, tuttavia, dovrai abilitare le notifiche degli eventi. Per informazioni, consulta [Imposta la notifica degli eventi per il posizionamento della sessione di gioco](#).
- (Facoltativo) Aggiungi componenti di FlexMatch matchmaking. Per informazioni, consulta la [guida per gli Amazon GameLift ServersFlexMatch sviluppatori](#).

Fase 4: Configura una flotta Anywhere con l'Amazon GameLift Serversagente

Fino a questo punto hai utilizzato dispositivi locali (registrati come Anywhere fleet computes) per testare e iterare i componenti del gioco. Il passaggio successivo consiste nel configurare il tipo di flotte necessarie per un sistema di produzione. Inizia con una flotta Anywhere e aggiungi l'Amazon GameLift ServersAgent per gestire alcune attività chiave di gestione degli host on-computing. Per ulteriori dettagli, consultare [Collabora con l'Amazon GameLift Serversagente](#).

- Scarica l'Amazon GameLift Serversagente e aggiungilo al pacchetto di installazione del server di gioco. Ottieni e crea il codice sorgente dell'agente, disponibile nel [GitHub repository dell'Amazon GameLift Serversagente](#). Posiziona il file JAR risultante eseguibile nella stessa directory dell'eseguibile della build del gioco.
- Se necessario, modificate lo script di avvio per l'agente. Assicurati che l'eseguibile dell'agente venga avviato non appena un computer inizia a funzionare. Consulta il file readme nel repository Agent per informazioni sull'installazione e l'esecuzione dell'agente sui computer di hosting. Il comando di avvio dovrebbe includere opzioni per specificare, come minimo, l'Anywhere fleet ID e Regione AWS una posizione personalizzata e un nome di calcolo.

L'agente gestisce automaticamente le seguenti attività per te, quindi se hai gestito queste attività con script, puoi rimuoverle:

- Chiamate RegisterCompute() per aggiungere l'elaborazione a una flotta Anywhere.

- Chiamate `GetComputeAuthToken()` per autenticare i server di gioco quando si connettono al Amazon GameLift Servers servizio. L'agente gestisce il recupero e l'aggiornamento del token di autenticazione, che può essere utilizzato da tutti i processi del server di gioco in esecuzione sul computer.
- Avvia nuovi processi del server sul computer in base a una serie di istruzioni di runtime.
- Crea una configurazione di runtime per i computer della tua flotta Anywhere. Specificate almeno il percorso di avvio per l'eseguibile del server di gioco. Puoi utilizzare la Amazon GameLift Servers console o la AWS CLI per creare o modificare le istruzioni di runtime per la flotta. L'agente esegue queste istruzioni e richiede periodicamente aggiornamenti al Amazon GameLift Servers servizio.
- Configura o modifica la coda delle sessioni di gioco in base alle tue esigenze. Crea una nuova coda (o aggiornane una esistente) e designa una destinazione per la flotta Anywhere schierata con l'agente. Amazon GameLift Servers
- Testa l'integrazione di Agent con le tue flotte Anywhere. Verifica che l'agente stia avviando correttamente i processi del server in base alla configurazione di runtime.

Fase 5: Creare una flotta gestita basata sul cloud

Crea una flotta EC2 Amazon GameLift Servers gestita per integrare la tua flotta Anywhere. Se configuri un ambiente di test basato sul cloud nella Fase 2 per accelerare lo sviluppo, pianifica di creare una flotta gestita dopo aver completato la maggior parte dello sviluppo e dei test del gioco. È necessaria una flotta completamente gestita per configurare e testare impostazioni aggiuntive come il ridimensionamento automatico della capacità.

- Package della build del tuo server di gioco e caricala su Amazon GameLift Servers. Crea uno script di installazione con i tuoi file di build, le dipendenze e il software di supporto. Puoi utilizzare lo stesso software di build sia con la tua flotta Anywhere che con quella gestita. Per informazioni, consulta [Crea un server di gioco costruito per Amazon GameLift Servers](#). Puoi caricare la tua build Amazon GameLift Servers utilizzando la console o la AWS CLI.

Prima di caricare la build, decidi in che Regione AWS vuoi creare la flotta gestita. Devi caricare la build nella stessa regione. Per ulteriori informazioni sulla scelta dell'ubicazione della flotta, consulta [Posizioni geografiche](#).

- Crea una flotta EC2 gestita. Puoi utilizzare la Amazon GameLift Servers console o la AWS CLI per creare una flotta gestita. Quando crei una flotta, inizia Amazon GameLift Servers immediatamente a distribuire il server di gioco creato per l'hosting. Puoi configurare molti aspetti di una flotta gestita.

Per le linee guida, consulta [Crea una EC2 flotta Amazon GameLift Servers gestita](#). Come minimo, procedi come segue:

- Assegna un nome alla flotta e specifica quale build di gioco caricata utilizzare.
- Scegli le istanze On-Demand per il tuo parco istanze e seleziona un tipo di istanza disponibile in base all'ubicazione del tuo parco istanze. Le flotte Spot sono un'opzione preziosa, ma richiedono un design e una configurazione aggiuntivi.
- Crea una configurazione di runtime con impostazioni simili a quelle utilizzate con la flotta Anywhere. Specificate almeno il percorso di avvio per l'eseguibile del server di gioco.
- Specificate le impostazioni delle porte per consentire al traffico in entrata di accedere ai server di gioco.
- Aggiungi la flotta gestita alla coda delle sessioni di gioco condivise. Aggiorna la coda dalla Fase 4 in modo che includa le destinazioni sia per la flotta gestita che per la flotta Anywhere schierata con l'agente. Amazon GameLift Servers
- Prova l'hosting dei giochi con le tue flotte gestite. A questo punto dovresti essere in grado di testare l'intero ciclo di hosting, con un client di gioco che richiede una sessione di gioco, ottiene informazioni di connessione e si connette correttamente a una sessione di gioco.

Fase 6: Personalizza le tue flotte

Mentre ti prepari per il lancio del gioco, dovrai perfezionare le tue soluzioni di hosting. Alcune delle decisioni da prendere in considerazione includono:

- Per le flotte Anywhere, automatizza il processo di avvio e spegnimento dei computer in base alle esigenze, inclusa l'installazione e l'esecuzione del software del server di gioco. Il riciclaggio dei computer è utile per garantire che vengano aggiornati regolarmente e lo spegnimento dei computer può far risparmiare sui costi quando non sono necessari.
- Se il tuo server di gioco deve comunicare altre AWS risorse, configura i ruoli IAM per gestire l'accesso. Per informazioni, consulta [Connect il server di gioco Amazon GameLift Servers ospitato ad altre AWS risorse](#).
- Determina dove vuoi posizionare geograficamente i server di gioco. Aggiungi postazioni remote alle tue flotte gestite. Per informazioni, consulta [Personalizzazioni delle risorse di hosting](#).
- Per le flotte gestite, prendi in considerazione l'utilizzo delle flotte Spot per risparmiare sui costi. Per informazioni, consulta [Riduci i costi di hosting dei giochi con le flotte Spot](#).
- Ottimizza le prestazioni della flotta selezionando le configurazioni delle risorse di calcolo, quindi configura le istruzioni di runtime per eseguire il numero ottimale di processi server per

elaborazione. Esegui questa operazione sia per le flotte Anywhere che per le flotte gestite. Per informazioni, consulta [Ottimizza la configurazione del runtime del server di gioco su server gestiti Amazon GameLift Servers](#).

- Sperimenta le opzioni di posizionamento delle sessioni di gioco per le flotte gestite, inclusa la personalizzazione delle impostazioni di prioritizzazione. Per informazioni, consulta [Personalizza una coda di sessioni di gioco](#).
- Per le flotte gestite, imposta la scalabilità automatica della capacità per soddisfare la domanda prevista dei giocatori. Per informazioni, consulta [Scalabilità della capacità di hosting di giochi con Amazon GameLift Servers](#).
- Per le flotte Anywhere, crea meccanismi per gestire la scalabilità manuale o automatizzata della capacità per soddisfare la domanda prevista dei giocatori.
- Progetta e implementa il failover su altre risorse, se necessario. Configura flotte di standby in altre Regioni AWS e modifica le code e la scalabilità automatica per gestire i failover, se necessario.
- Configura gli strumenti di osservabilità dell'hosting, tra cui analisi e registrazione. Per informazioni, consulta [Monitoraggio di Amazon GameLift Servers](#). Crea gruppi di metrici per aggregare le analisi per tutte le tue risorse di hosting.
- Automatizza la distribuzione utilizzando [Infrastructure as Code \(IaC\)](#). Per informazioni, consulta [Gestisci le risorse di Amazon GameLift Servers hosting utilizzando CloudFormation](#).

Amazon GameLift Servers supporta l'uso di AWS CloudFormation modelli per qualsiasi configurazione specifica dell'implementazione. È inoltre possibile utilizzare il per definire le risorse. AWS Cloud Development Kit (AWS CDK) Amazon GameLift Servers Per ulteriori informazioni su AWS CDK, consulta la [Guida per AWS Cloud Development Kit \(AWS CDK\) gli sviluppatori](#).

Per gestire l'implementazione degli CloudFormation stack, consigliamo di utilizzare strumenti e servizi di integrazione continua e distribuzione continua (CI/CD) come. AWS CodePipeline Questi strumenti ti aiutano a implementarlo automaticamente o con l'approvazione ogni volta che crei un file binario per server di gioco. Con CI/CD uno strumento o un servizio, la distribuzione delle risorse per una nuova versione del server di gioco può essere simile a questa:

- Crea e testa il file binario del tuo server di gioco.
- Carica il file binario su Amazon GameLift Servers.
- Implementa nuove flotte con la nuova build.
- Aggiungi le nuove flotte alla coda delle sessioni di gioco e rimuovi le flotte con la versione di build precedente.

- Quando le flotte della build precedente non ospitano più sessioni di gioco attive, elimina le CloudFormation pile di quelle flotte.

Prepara un gioco da ospitare con Amazon GameLift Servers

Prima di poter distribuire il gioco per l'hosting su Amazon GameLift Servers, devi preparare sia il server di gioco che i componenti del client di gioco. Ciò comporta l'integrazione del server di gioco con Amazon GameLift Servers Server SDK e la creazione di un servizio di backend che gestisca l'autenticazione dei giocatori e la gestione delle sessioni di gioco.

L'integrazione con Server SDK consente al server di gioco di comunicare con il Amazon GameLift Servers servizio, segnalarne lo stato di salute e gestire le sessioni di gioco. Il servizio di backend funge da intermediario tra i client di gioco e il Amazon GameLift Servers servizio, gestendo le richieste dei giocatori e coordinando il posizionamento delle sessioni di gioco.

Argomenti

- [Prepara il tuo gioco Unreal o Unity con il plugin Amazon GameLift Servers](#)
- [Integra un server di gioco con Amazon GameLift Servers](#)
- [Package di un server di gioco costruito per la distribuzione](#)
- [client/server Interazioni di gioco con Amazon GameLift Servers](#)

Prepara il tuo gioco Unreal o Unity con il plugin Amazon GameLift Servers

Il Amazon GameLift Servers plugin è un componente aggiuntivo completo del tuo motore di gioco Unreal o Unity. Ti guida attraverso i passaggi di base con cui distribuire il gioco per l'hosting. Amazon GameLift Servers Con il set di strumenti e i flussi di lavoro del plug-in, puoi lavorare nell'ambiente di sviluppo del tuo motore di gioco per preparare il server di gioco per l'hosting, configurare l'hosting su un computer locale per i test, creare un semplice servizio di backend e distribuire il server di gioco su un hosting gestito basato su cloud.

Usa il plug-in per provare a utilizzare Amazon GameLift Servers e ottenere rapidamente una soluzione di hosting di giochi attiva e funzionante. Puoi lavorare con risorse di gioco di esempio o con il tuo progetto di gioco. Il plugin automatizza una serie di passaggi in modo da poter creare rapidamente una soluzione di lavoro semplice. Una volta completati i flussi di lavoro guidati del plug-in, sarai in grado di connettere un client di gioco a sessioni di gioco ospitate dal vivo. Amazon

GameLift Servers Dopo aver usato il plugin per creare una soluzione di hosting semplice, puoi personalizzare la soluzione per soddisfare le esigenze del tuo gioco.

Il plugin è disponibile per i seguenti motori di gioco:

- Unreal Engine
- Unity

Il plugin include questi componenti per ogni motore di gioco:

- Moduli plug-in per l'editor del motore di gioco. Quando il plugin è installato, un nuovo pulsante del menu principale consente di accedere alle Amazon GameLift Servers funzionalità.
- Librerie per l'API Amazon GameLift Servers di servizio con funzionalità lato client.
- Librerie per l'SDK Amazon GameLift Servers del server (versione 5).
- Risorse di esempio da utilizzare per testare l'integrazione di un server.
- Configurazioni modificabili, sotto forma di CloudFormation modelli, che definiscono la soluzione del server di gioco.

Argomenti

- [Workflow dei plugin](#)
- [Amazon GameLift Serversplugin per Unreal Engine](#)
- [Amazon GameLift Serversplugin per Unity \(server SDK 5.x\)](#)
- [Amazon GameLift Serversplugin per Unity per server SDK 4](#)

Workflow dei plugin

I passaggi seguenti descrivono un percorso tipico di preparazione e implementazione del progetto di gioco. Amazon GameLift Servers Puoi completare questi passaggi utilizzando l'editor del motore di gioco e il codice del gioco.

1. Crea un profilo utente che si colleghi all'utente del tuo AWS account e fornisca le credenziali di accesso con le autorizzazioni da utilizzare. Amazon GameLift Servers
2. Imposta AWS le risorse correlate utilizzate dal plug-in nella soluzione di hosting (denominata «bootstrap»).

3. Aggiungi il codice server al tuo progetto per stabilire la comunicazione tra un server di gioco in esecuzione e il Amazon GameLift Servers servizio.
4. Aggiungi un codice client al tuo progetto che consenta ai client di gioco di Amazon GameLift Servers inviare richieste per avviare nuove sessioni di gioco e quindi connettersi ad esse.
5. Usa il flusso di lavoro Anywhere per configurare la tua workstation locale come computer Anywhere e ospitare il tuo server di gioco. Avvia il server e il client di gioco localmente tramite il plug-in, connettiti a una sessione di gioco e verifica l'integrazione.
6. Usa il EC2 flusso di lavoro gestito per caricare il tuo server di gioco Amazon GameLift Servers e implementare una soluzione di hosting cloud semplice ma completa. Avvia il client di gioco localmente tramite il plug-in, richiedi una sessione di gioco, connettiti ad esso e gioca.

Quando lavori con il plugin, creerai e utilizzerai AWS risorse. Queste azioni potrebbero comportare addebiti sull' AWS account in uso. Se non l'hai mai fatto AWS, queste azioni potrebbero essere incluse nel piano [AWS gratuito](#).

Amazon GameLift Serversplugin per Unreal Engine

Questo plugin aggiunge l'SDK e gli strumenti del server Amazon GameLift Servers C++ all'editor UE. Utilizza i flussi di lavoro guidati dell'interfaccia utente per integrare la funzionalità SDK del server nel tuo progetto di gioco e implementare una soluzione di Amazon GameLift Servers hosting per il tuo server di gioco.

Con il plug-in, puoi creare una soluzione di hosting funzionante di base e quindi ottimizzarla e personalizzarla secondo necessità. Configura una flotta Amazon GameLift Servers Anywhere con la tua workstation locale come host. Per l'hosting su cloud con EC2 gestito o flotte di container gestite, implementa il tuo server di gioco con una soluzione completa per gestire le richieste di sessioni di gioco e le connessioni dei client.

Argomenti

- [Installa il plugin per il tuo progetto di gioco Unreal](#)
- [Passaggi successivi: personalizza la tua soluzione di hosting di giochi](#)
- [Plugin per Unreal: configura un profilo AWS utente](#)
- [Plugin per Unreal: integra il tuo codice di gioco](#)
- [Plugin per Unreal: ospita il tuo gioco localmente con Anywhere Amazon GameLift Servers](#)
- [Plugin per Unreal: distribuisce il tuo gioco su una flotta EC2 gestita](#)
- [Plugin per Unreal: distribuisce il gioco su una flotta di container gestita](#)

Installa il plugin per il tuo progetto di gioco Unreal

[Scarica il Amazon GameLift Servers plugin per Unreal Engine da GitHub](#)

Consulta il readme del GitHub repository per informazioni su come installare il plugin nel tuo Unreal Editor per un progetto di gioco.

Il plugin include questi componenti:

- Moduli plug-in per l'editor UE. Quando il plugin è installato, un nuovo pulsante del menu principale consente di accedere alle Amazon GameLift Servers funzionalità.
- Librerie C++ per l'API del Amazon GameLift Servers servizio. Utilizza la funzionalità API in un servizio di backend lato client per aiutare i client di gioco a richiedere sessioni di gioco e send/retrieve informazioni sulle sessioni di gioco.
- Librerie Unreal per l'SDK del Amazon GameLift Servers server (versione 5). Utilizza l'SDK del server nel codice del server di gioco per gestire la comunicazione tra i processi del server di gioco ospitato e il servizio. Amazon GameLift Servers
- Contenuti da testare, tra cui una mappa di gioco iniziale e due mappe di test con progetti di base ed elementi dell'interfaccia utente da utilizzare per testare l'integrazione di un server.
- Configurazioni modificabili, sotto forma di CloudFormation modelli, utilizzate dal plug-in per distribuire il server di gioco per l'hosting.

Questo plugin utilizza AWS CloudFormation modelli per implementare soluzioni di hosting per scenari di gioco comuni. Utilizza queste soluzioni così come fornite o personalizzate secondo necessità per i tuoi giochi.

Passaggi successivi: personalizza la tua soluzione di hosting di giochi

L'utilizzo dei flussi di lavoro guidati del plug-in è un buon modo per iniziare a utilizzare rapidamente una soluzione di Amazon GameLift Servers hosting. Con il plug-in, puoi configurare le versioni di base di ciascuno dei componenti della tua soluzione.

Quando sei pronto, puoi sfruttare questa soluzione di base personalizzando ogni componente e perfezionandola mentre ti prepari al lancio del gioco. Prendi in considerazione queste opzioni:

- Modifica le flotte e le configurazioni della flotta. Per informazioni, consulta [Personalizzazioni delle risorse di hosting](#).
- Personalizza la configurazione della coda delle sessioni di gioco. Vedi: [Personalizza una coda di sessioni di gioco](#)

- Aggiungi funzionalità al server di gioco e al client di gioco. Consulta [Integra un server di gioco con Amazon GameLift Servers](#) e [Integra la funzionalità del client di Amazon GameLift Servers gioco](#).
- Personalizza il tuo servizio di backend. Per informazioni, consulta [Crea un servizio di backend per Amazon GameLift Servers](#).
- Imposta il ridimensionamento automatico della capacità per soddisfare la domanda prevista dei giocatori. Per informazioni, consulta [Scalabilità della capacità di hosting di giochi con Amazon GameLift Servers](#).
- Configura strumenti di osservabilità dell'hosting, tra cui analisi e registrazione. Per informazioni, consulta [Monitoraggio di Amazon GameLift Servers](#).
- Automatizza la distribuzione utilizzando [Infrastructure as Code \(IaC\)](#). I flussi di lavoro guidati del plug-in per le soluzioni gestite utilizzano modelli. AWS CloudFormation Puoi personalizzarli secondo necessità. Per informazioni, consulta [Gestisci le risorse di Amazon GameLift Servers hosting utilizzando CloudFormation](#).

Argomenti

- [Plugin per Unreal: configura un profilo AWS utente](#)
- [Plugin per Unreal: integra il tuo codice di gioco](#)
- [Plugin per Unreal: ospita il tuo gioco localmente con Anywhere Amazon GameLift Servers](#)
- [Plugin per Unreal: distribuisce il tuo gioco su una flotta EC2 gestita](#)
- [Plugin per Unreal: distribuisce il gioco su una flotta di container gestita](#)

Plugin per Unreal: configura un profilo AWS utente

Dopo aver installato il plugin, configura un profilo utente con un AWS account valido. È possibile mantenere più profili nel plug-in, ma è possibile selezionare un solo profilo alla volta. Ogni volta che lavori nel plugin, seleziona un profilo da usare. Ogni pagina del flusso di lavoro mostra il profilo attualmente selezionato.

Il mantenimento di più profili consente di passare da una distribuzione di hosting all'altra. Ad esempio, è possibile configurare profili che utilizzano lo stesso AWS account ma vengono distribuiti in regioni diverse. AWS Oppure puoi configurare profili con AWS account o utenti e set di autorizzazioni diversi.

 Note

Se hai installato la AWS CLI sulla tua workstation e hai già un profilo configurato, il Amazon GameLift Servers plugin lo rileverà e lo elencherà come profilo esistente. Il plugin seleziona automaticamente qualsiasi profilo denominato. [default] È possibile utilizzare un profilo esistente o crearne uno nuovo.

Tutti i profili devono essere avviati per configurare alcune AWS risorse necessarie nell'account utente.

Per gestire i tuoi profili AWS

1. Nella barra degli strumenti principale dell'editor di Unreal, scegli il Amazon GameLift Servers menu e seleziona Credenziali di AWS accesso. Questa azione apre il Amazon GameLift Servers plugin alla pagina Configura il tuo profilo utente.
2. Utilizza i pulsanti per creare un nuovo AWS account o configurare un profilo utente per un AWS account che già possiedi.
3. Se non disponi già di un profilo utente, ti verrà richiesto di inserire i dettagli del profilo e creare un nuovo profilo. Inserisci le informazioni che seguono:
 - Un AWS account. Se crei un nuovo AWS account, usa il link Console di gestione AWS e segui le istruzioni. Vedi [Creare un AWS account](#) per maggiori dettagli.
 - Un AWS utente con autorizzazioni all'uso Amazon GameLift Servers e altri AWS servizi richiesti. Consulta [Configurare un account AWS utente](#) le istruzioni sulla configurazione di un utente AWS Identity and Access Management (IAM) con Amazon GameLift Servers autorizzazioni e accesso programmatico con credenziali a lungo termine.
 - Credenziali per il tuo utente. AWS Queste credenziali sono costituite da un ID della chiave di AWS accesso e da una chiave AWS segreta. Per maggiori dettagli, consulta [Ottenere le chiavi di accesso](#).
 - AWS regione. Questa è una posizione geografica in cui desideri creare AWS le tue risorse per l'hosting. Durante lo sviluppo, ti consigliamo di utilizzare un'area vicina alla tua posizione fisica. Scegli una regione dall'elenco delle [AWS regioni supportate](#).
4. Se il plugin rileva profili esistenti, visualizza un elenco di profili disponibili. Seleziona un profilo esistente dall'elenco o scegli Aggiungi un altro profilo per crearne uno nuovo.

Avvia un profilo utente

Tutti i profili devono essere avviati per essere utilizzati con il plugin. Amazon GameLift Servers Il bootstrap crea un bucket Amazon S3 specifico per il profilo. Viene utilizzato per archiviare configurazioni di progetto, creare artefatti e altre dipendenze. I bucket non vengono condivisi tra altri profili.

Il bootstrap implica la creazione di nuove AWS risorse e potrebbe comportare dei costi.

Per avviare il tuo profilo:

1. Nella pagina Credenziali di AWS accesso, controlla lo stato di bootstrap del profilo utente che desideri utilizzare. Se lo stato di bootstrap del profilo è «Inattivo» e non è presente alcun bucket S3 nell'elenco, devi avviare il profilo.
2. Seleziona il profilo che desideri utilizzare e scegli il profilo Bootstrap.
3. Attendi che lo stato del bootstrap passi a «Attivo». Ciò può richiedere alcuni minuti.

Plugin per Unreal: integra il tuo codice di gioco

Prima di poter distribuire il server di gioco su una flotta, devi apportare una serie di aggiornamenti al codice di gioco e impacchettare i componenti di gioco da utilizzare con il Amazon GameLift Servers servizio.

Questo argomento illustra i passaggi per eseguire un'integrazione minima. Per l'integrazione con il server, usa l'esempio di codice fornito per aggiornare la modalità di gioco del progetto.

- [Imposta gli obiettivi di costruzione e le regole del modulo](#)
- [Aggiorna il codice del server di gioco](#)
- [Integra la mappa di gioco del tuo client](#)
- [Package dei componenti del gioco](#)

Imposta gli obiettivi di costruzione e le regole del modulo

Modifica i file di progetto del gioco per generare correttamente i componenti di compilazione da utilizzare con Amazon GameLift Servers.

Per aggiungere obiettivi di build per client e server:

1. Apri i file di codice del progetto di gioco e individua il file del `.../Games/[your application name]Source/[your application name]Target.cs` file. Esempio: `.../Source/GameLiftUnrealAppTarget.cs`. (Se usi Visual Studio, apri il `.sln` file del progetto).
2. Copia questo file per creare due nuovi file di destinazione nella `Source/` directory.
 - Target client: rinomina il nuovo file in `[your application name]Client.Target.cs`. Modifica il contenuto per aggiornare i valori del nome della classe e del tipo di destinazione, come illustrato nel seguente codice di esempio:

```
using UnrealBuildTool;
using System.Collections.Generic;

public class GameLiftUnrealAppClientTarget : TargetRules
{
    public GameLiftUnrealAppClientTarget ( TargetInfo Target ) : base
( Target )
    {
        Type = TargetType.Client;
        DefaultBuildSettings = BuildSettingsVersion.V2;
        IncludeOrderVersion = EngineIncludeOrderVersion.Unreal5_1;
        ExtraModuleNames.Add( "GameLiftUnrealApp" );
    }
}
```

- Destinazione del server: rinomina il nuovo file in `[your application name]Server.Target.cs`. Modifica il contenuto per aggiornare i valori del nome della classe e del tipo di destinazione, come illustrato nel seguente codice di esempio:

```
using UnrealBuildTool;
using System.Collections.Generic;

public class GameLiftUnrealAppServerTarget : TargetRules
{
    public GameLiftUnrealAppServerTarget ( TargetInfo Target ) : base
( Target )
    {
        Type = TargetType.Server;
        DefaultBuildSettings = BuildSettingsVersion.V2;
        IncludeOrderVersion = EngineIncludeOrderVersion.Unreal5_1;
        ExtraModuleNames.Add( "GameLiftUnrealApp" );
    }
}
```



```
}  
}
```

3. Rigenera i file di progetto. Se usi Visual Studio, puoi fare clic con il pulsante destro del mouse sul .uproject file del progetto di gioco e selezionare Genera file di progetto di Visual Studio.

Per aggiornare le regole del modulo del progetto di gioco:

Aggiorna le regole del modulo del progetto di gioco per fare in modo che dipenda dal plugin.

1. Apri i file di codice del progetto di gioco e individua il ../Games/[*your application name*]/Source/[*your application name*].Build.cs file. Esempio: ../Source/GameLiftUnrealApp.Build.cs. (Se usi Visual Studio, apri il .sln file del progetto).
2. Individua la ModuleRules classe e aggiorna come illustrato nel seguente codice di esempio:

```
using UnrealBuildTool;  
  
public class GameLiftUnrealApp : ModuleRules  
{  
    public GameLiftUnrealApp ( ReadOnlyTargetRules Target ) : base ( Target )  
    {  
        PCHUsage = PCHUsageMode.UseExplicitOrSharedPCHs;  
        PublicDependencyModuleNames.AddRange( new string[] { "Core",  
            "CoreUObject", "Engine", "InputCore", "HeadMountedDisplay",  
            "EnhancedInput" });  
        // Add the following section  
        if (Target.Type == TargetType.Server)  
        {  
            PublicDependencyModuleNames.Add("GameLiftServerSDK");  
        }  
        else  
        {  
            PublicDefinitions.Add("WITH_GAMELIFT=0");  
        }  
        bEnableExceptions = true;  
    }  
}
```

3. Dopo aver creato i nuovi file di destinazione e modificato le regole del modulo, ricostruisci il progetto del gioco.

Aggiorna il codice del server di gioco

Aggiorna il codice del server di gioco per abilitare la comunicazione tra un processo del server di gioco e il Amazon GameLift Servers servizio. Il server di gioco deve essere in grado di rispondere alle richieste provenienti da Amazon GameLift Servers, ad esempio, dall'avvio e dall'interruzione di nuove sessioni di gioco.

Per aggiungere il codice del server per Amazon GameLift Servers

1. Nel tuo editor di codice, apri il file solution (.sln) per il tuo progetto di gioco, che di solito si trova nella cartella principale del progetto. Ad esempio: `GameLiftUnrealApp.sln`.
2. Con la soluzione aperta, individua il file di intestazione della modalità gioco del progetto: `[project-name]GameMode.h` file. Ad esempio: `GameLiftUnrealAppGameMode.h`.
3. Modifica il file di intestazione per allinearli al codice seguente. Assicurati di sostituire "GameLiftServer" con il nome del tuo progetto. Questi aggiornamenti sono specifici per il server di gioco; ti consigliamo di creare una copia di backup dei file delle modalità di gioco originali da utilizzare con il tuo client.

Esempio di codice GameMode.h

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0

#pragma once

#include "CoreMinimal.h"
#include "GameFramework/GameModeBase.h"
#include "GameLiftUnrealAppGameMode.generated.h"

struct FProcessParameters;

DECLARE_LOG_CATEGORY_EXTERN(GameServerLog, Log, All);

UCLASS(minimalapi)
class AGameLiftUnrealAppGameMode : public AGameModeBase
{
    GENERATED_BODY()

public:
    AGameLiftUnrealAppGameMode();
```

```
protected:
    virtual void BeginPlay() override;

private:
    void InitGameLift();

private:
    TSharedPtr<FProcessParameters> ProcessParameters;
};
```

- Apri il [project-name]GameMode.cpp file sorgente correlato (ad esempio). GameLiftUnrealAppGameMode.cpp Modificate il codice per allinearli al codice di esempio seguente. Assicurati di sostituire "GameLiftUnrealApp" con il nome del tuo progetto. Questi aggiornamenti sono specifici per il server di gioco; ti consigliamo di creare una copia di backup del file originale da utilizzare con il tuo client.

Il codice di esempio seguente mostra come aggiungere gli elementi minimi richiesti per l'integrazione del server con Amazon GameLift Servers:

- Inizializza un client Amazon GameLift Servers API. La InitSDK() chiamata con i parametri del server è necessaria per una flotta Amazon GameLift Servers Anywhere. Quando ti connetti a una flotta Anywhere, il plug-in memorizza i parametri del server come argomenti della console. Il codice di esempio può accedere ai valori in fase di esecuzione.
- Implementa le funzioni di callback richieste per rispondere alle richieste del Amazon GameLift Servers servizio, tra cui OnStartGameSessionOnProcessTerminate, eonHealthCheck.
- Chiama ProcessReady() con una porta designata per avvisare il Amazon GameLift Servers servizio quando è pronto per ospitare sessioni di gioco.

Esempio di codice del server di gioco

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0

#include "GameLiftUnrealAppGameMode.h"

#include "UObject/ConstructorHelpers.h"
#include "Kismet/GameplayStatics.h"
```

```

#if WITH_GAMELIFT
#include "GameLiftServerSDK.h"
#include "GameLiftServerSDKModels.h"
#endif

#include "GenericPlatform/GenericPlatformOutputDevices.h"

DEFINE_LOG_CATEGORY(GameServerLog);

AGameLiftUnrealAppGameMode::AGameLiftUnrealAppGameMode() :
    ProcessParameters(nullptr)
{
    // Set default pawn class to our Blueprinted character
    static ConstructorHelpers::FClassFinder<APawn> PlayerPawnBPClass(TEXT("/Game/
ThirdPerson/Blueprints/BP_ThirdPersonCharacter"));

    if (PlayerPawnBPClass.Class != NULL)
    {
        DefaultPawnClass = PlayerPawnBPClass.Class;
    }

    UE_LOG(GameServerLog, Log, TEXT("Initializing AGameLiftUnrealAppGameMode..."));
}

void AGameLiftUnrealAppGameMode::BeginPlay()
{
    Super::BeginPlay();

#if WITH_GAMELIFT
    InitGameLift();
#endif
}

void AGameLiftUnrealAppGameMode::InitGameLift()
{
#if WITH_GAMELIFT
    UE_LOG(GameServerLog, Log, TEXT("Calling InitGameLift..."));

    // Getting the module first.
    FGameLiftServerSDKModule* GameLiftSdkModule =
    &FModuleManager::LoadModuleChecked<FGameLiftServerSDKModule>(FName("GameLiftServerSDK"));

    //Define the server parameters for a GameLift Anywhere fleet. These are not needed
    for a GameLift managed EC2 fleet.

```

```
FServerParameters ServerParametersForAnywhere;

bool bIsAnywhereActive = false;
if (FParse::Param(FCommandLine::Get(), TEXT("glAnywhere")))
{
    bIsAnywhereActive = true;
}

if (bIsAnywhereActive)
{
    UE_LOG(GameServerLog, Log, TEXT("Configuring server parameters for
Anywhere..."));

    // If GameLift Anywhere is enabled, parse command line arguments and pass them
in the ServerParameters object.
    FString glAnywhereWebSocketUrl = "";
    if (FParse::Value(FCommandLine::Get(), TEXT("glAnywhereWebSocketUrl="),
glAnywhereWebSocketUrl))
    {
        ServerParametersForAnywhere.m_webSocketUrl =
TCHAR_TO_UTF8(*glAnywhereWebSocketUrl);
    }

    FString glAnywhereFleetId = "";
    if (FParse::Value(FCommandLine::Get(), TEXT("glAnywhereFleetId="),
glAnywhereFleetId))
    {
        ServerParametersForAnywhere.m_fleetId = TCHAR_TO_UTF8(*glAnywhereFleetId);
    }

    FString glAnywhereProcessId = "";
    if (FParse::Value(FCommandLine::Get(), TEXT("glAnywhereProcessId="),
glAnywhereProcessId))
    {
        ServerParametersForAnywhere.m_processId =
TCHAR_TO_UTF8(*glAnywhereProcessId);
    }
    else
    {
        // If no ProcessId is passed as a command line argument, generate a
randomized unique string.
        FString TimeString = FString::FromInt(std::time(nullptr));
        FString ProcessId = "ProcessId_" + TimeString;
        ServerParametersForAnywhere.m_processId = TCHAR_TO_UTF8(*ProcessId);
    }
}
```

```
    }

    FString glAnywhereHostId = "";
    if (FParse::Value(FCommandLine::Get(), TEXT("glAnywhereHostId="),
glAnywhereHostId))
    {
        ServerParametersForAnywhere.m_hostId = TCHAR_TO_UTF8(*glAnywhereHostId);
    }

    FString glAnywhereAuthToken = "";
    if (FParse::Value(FCommandLine::Get(), TEXT("glAnywhereAuthToken="),
glAnywhereAuthToken))
    {
        ServerParametersForAnywhere.m_authToken =
TCHAR_TO_UTF8(*glAnywhereAuthToken);
    }

    FString glAnywhereAwsRegion = "";
    if (FParse::Value(FCommandLine::Get(), TEXT("glAnywhereAwsRegion="),
glAnywhereAwsRegion))
    {
        ServerParametersForAnywhere.m_awsRegion =
TCHAR_TO_UTF8(*glAnywhereAwsRegion);
    }

    FString glAnywhereAccessKey = "";
    if (FParse::Value(FCommandLine::Get(), TEXT("glAnywhereAccessKey="),
glAnywhereAccessKey))
    {
        ServerParametersForAnywhere.m_accessKey =
TCHAR_TO_UTF8(*glAnywhereAccessKey);
    }

    FString glAnywhereSecretKey = "";
    if (FParse::Value(FCommandLine::Get(), TEXT("glAnywhereSecretKey="),
glAnywhereSecretKey))
    {
        ServerParametersForAnywhere.m_secretKey =
TCHAR_TO_UTF8(*glAnywhereSecretKey);
    }

    FString glAnywhereSessionToken = "";
    if (FParse::Value(FCommandLine::Get(), TEXT("glAnywhereSessionToken="),
glAnywhereSessionToken))
```

```

    {
        ServerParametersForAnywhere.m_sessionToken =
TCHAR_TO_UTF8(*glAnywhereSessionToken);
    }

    UE_LOG(GameServerLog, SetColor, TEXT("%s"), COLOR_YELLOW);
    UE_LOG(GameServerLog, Log, TEXT(">>>> WebSocket URL: %s"),
*ServerParametersForAnywhere.m_webSocketUrl);
    UE_LOG(GameServerLog, Log, TEXT(">>>> Fleet ID: %s"),
*ServerParametersForAnywhere.m_fleetId);
    UE_LOG(GameServerLog, Log, TEXT(">>>> Process ID: %s"),
*ServerParametersForAnywhere.m_processId);
    UE_LOG(GameServerLog, Log, TEXT(">>>> Host ID (Compute Name): %s"),
*ServerParametersForAnywhere.m_hostId);
    UE_LOG(GameServerLog, Log, TEXT(">>>> Auth Token: %s"),
*ServerParametersForAnywhere.m_authToken);
    UE_LOG(GameServerLog, Log, TEXT(">>>> Aws Region: %s"),
*ServerParametersForAnywhere.m_awsRegion);
    UE_LOG(GameServerLog, Log, TEXT(">>>> Access Key: %s"),
*ServerParametersForAnywhere.m_accessKey);
    UE_LOG(GameServerLog, Log, TEXT(">>>> Secret Key: %s"),
*ServerParametersForAnywhere.m_secretKey);
    UE_LOG(GameServerLog, Log, TEXT(">>>> Session Token: %s"),
*ServerParametersForAnywhere.m_sessionToken);
    UE_LOG(GameServerLog, SetColor, TEXT("%s"), COLOR_NONE);
}

UE_LOG(GameServerLog, Log, TEXT("Initializing the GameLift Server..."));

//InitSDK will establish a local connection with GameLift's agent to enable further
communication.
FGameLiftGenericOutcome InitSdkOutcome = GameLiftSdkModule-
>InitSDK(ServerParametersForAnywhere);
if (InitSdkOutcome.IsSuccess())
{
    UE_LOG(GameServerLog, SetColor, TEXT("%s"), COLOR_GREEN);
    UE_LOG(GameServerLog, Log, TEXT("GameLift InitSDK succeeded!"));
    UE_LOG(GameServerLog, SetColor, TEXT("%s"), COLOR_NONE);
}
else
{
    UE_LOG(GameServerLog, SetColor, TEXT("%s"), COLOR_RED);
    UE_LOG(GameServerLog, Log, TEXT("ERROR: InitSDK failed : ("));
    FGameLiftError GameLiftError = InitSdkOutcome.GetError();

```

```

    UE_LOG(GameServerLog, Log, TEXT("ERROR: %s"), *GameLiftError.m_errorMessage);
    UE_LOG(GameServerLog, SetColor, TEXT("%s"), COLOR_NONE);
    return;
}

ProcessParameters = MakeShared<FProcessParameters>();

//When a game session is created, Amazon GameLift Servers sends an activation
request to the game server and passes along the game session object containing game
properties and other settings.
//Here is where a game server should take action based on the game session object.
//Once the game server is ready to receive incoming player connections, it should
invoke GameLiftServerAPI.ActivateGameSession()
ProcessParameters->OnStartGameSession.BindLambda([=]
(Aws::GameLift::Server::Model::GameSession InGameSession)
{
    FString GameSessionId = FString(InGameSession.GetGameSessionId());
    UE_LOG(GameServerLog, Log, TEXT("GameSession Initializing: %s"),
*GameSessionId);
    GameLiftSdkModule->ActivateGameSession();
}));

//OnProcessTerminate callback. Amazon GameLift Servers will invoke this callback
before shutting down an instance hosting this game server.
//It gives this game server a chance to save its state, communicate with services,
etc., before being shut down.
//In this case, we simply tell Amazon GameLift Servers we are indeed going to
shutdown.
ProcessParameters->OnTerminate.BindLambda([=]()
{
    UE_LOG(GameServerLog, Log, TEXT("Game Server Process is terminating"));
    // First call ProcessEnding()
    FGameLiftGenericOutcome processEndingOutcome = GameLiftSdkModule-
>ProcessEnding();
    // Then call Destroy() to free the SDK from memory
    FGameLiftGenericOutcome destroyOutcome = GameLiftSdkModule->Destroy();
    // Exit the process with success or failure
    if (processEndingOutcome.IsSuccess() && destroyOutcome.IsSuccess()) {
        UE_LOG(GameServerLog, Log, TEXT("Server process ending successfully"));
    }
    else {
        if (!processEndingOutcome.IsSuccess()) {
            const FGameLiftError& error = processEndingOutcome.GetError();

```



```

        UE_LOG(GameServerLog, Error, TEXT("ProcessEnding() failed. Error:
%s"),
        error.m_errorMessage.IsEmpty() ? TEXT("Unknown error") :
*error.m_errorMessage);
    }
    if (!destroyOutcome.IsSuccess()) {
        const FGameLiftError& error = destroyOutcome.GetError();
        UE_LOG(GameServerLog, Error, TEXT("Destroy() failed. Error: %s"),
        error.m_errorMessage.IsEmpty() ? TEXT("Unknown error") :
*error.m_errorMessage);
    }
}

});

//This is the HealthCheck callback.
//Amazon GameLift Servers will invoke this callback every 60 seconds or so.
//Here, a game server might want to check the health of dependencies and such.
//Simply return true if healthy, false otherwise.
//The game server has 60 seconds to respond with its health status. Amazon GameLift
Servers will default to 'false' if the game server doesn't respond in time.
//In this case, we're always healthy!
ProcessParameters->OnHealthCheck.BindLambda([]()
{
    UE_LOG(GameServerLog, Log, TEXT("Performing Health Check"));
    return true;
});

//GameServer.exe -port=7777 LOG=server.mylog
ProcessParameters->port = FURL::UrlConfig.DefaultPort;
TArray<FString> CommandLineTokens;
TArray<FString> CommandLineSwitches;

FCommandLine::Parse(FCommandLine::Get(), CommandLineTokens, CommandLineSwitches);

for (FString SwitchStr : CommandLineSwitches)
{
    FString Key;
    FString Value;

    if (SwitchStr.Split("=", &Key, &Value))
    {
        if (Key.Equals("port"))
        {
            ProcessParameters->port = FCString::Atoi(*Value);

```

```

    }
}

//Here, the game server tells Amazon GameLift Servers where to find game session
log files.
//At the end of a game session, Amazon GameLift Servers uploads everything in the
specified
//location and stores it in the cloud for access later.
TArray<FString> Logfiles;
Logfiles.Add(TEXT("GameLiftUnrealApp/Saved/Logs/server.log"));
ProcessParameters->logParameters = Logfiles;

//The game server calls ProcessReady() to tell Amazon GameLift Servers it's ready
to host game sessions.
UE_LOG(GameServerLog, Log, TEXT("Calling Process Ready..."));
FGameLiftGenericOutcome ProcessReadyOutcome = GameLiftSdkModule-
>ProcessReady(*ProcessParameters);

if (ProcessReadyOutcome.IsSuccess())
{
    UE_LOG(GameServerLog, SetColor, TEXT("%s"), COLOR_GREEN);
    UE_LOG(GameServerLog, Log, TEXT("Process Ready!"));
    UE_LOG(GameServerLog, SetColor, TEXT("%s"), COLOR_NONE);
}
else
{
    UE_LOG(GameServerLog, SetColor, TEXT("%s"), COLOR_RED);
    UE_LOG(GameServerLog, Log, TEXT("ERROR: Process Ready Failed!"));
    FGameLiftError ProcessReadyError = ProcessReadyOutcome.GetError();
    UE_LOG(GameServerLog, Log, TEXT("ERROR: %s"),
*ProcessReadyError.m_errorMessage);
    UE_LOG(GameServerLog, SetColor, TEXT("%s"), COLOR_NONE);
}

UE_LOG(GameServerLog, Log, TEXT("InitGameLift completed!"));
#endif
}

```

Integra la mappa di gioco del tuo client

La mappa di gioco di avvio contiene la logica del progetto e gli elementi dell'interfaccia utente che includono già il codice di base per richiedere sessioni di gioco e utilizzare le informazioni di

connessione per connettersi a una sessione di gioco. Puoi usare la mappa così com'è o modificarla secondo necessità. Usa la mappa di gioco di avvio con altre risorse di gioco, come il modello di progetto Third Person fornito da Unreal Engine. Queste risorse sono disponibili in Content Browser. Puoi usarle per testare i flussi di lavoro di distribuzione del plugin o come guida per creare un servizio di backend personalizzato per il tuo gioco.

La mappa di avvio presenta le seguenti caratteristiche:

- Include la logica sia per una flotta Anywhere che per una flotta EC2 gestita. Quando gestisci il tuo client, puoi scegliere di connetterti a entrambe le flotte.
- Le funzionalità del client includono trovare una sessione di gioco (`SearchGameSessions()`), creare una nuova sessione di gioco (`CreateGameSession()`) e partecipare direttamente a una sessione di gioco.
- Ottiene un ID giocatore univoco dal pool di utenti Amazon Cognito del tuo progetto (fa parte di una soluzione distribuita Anywhere).

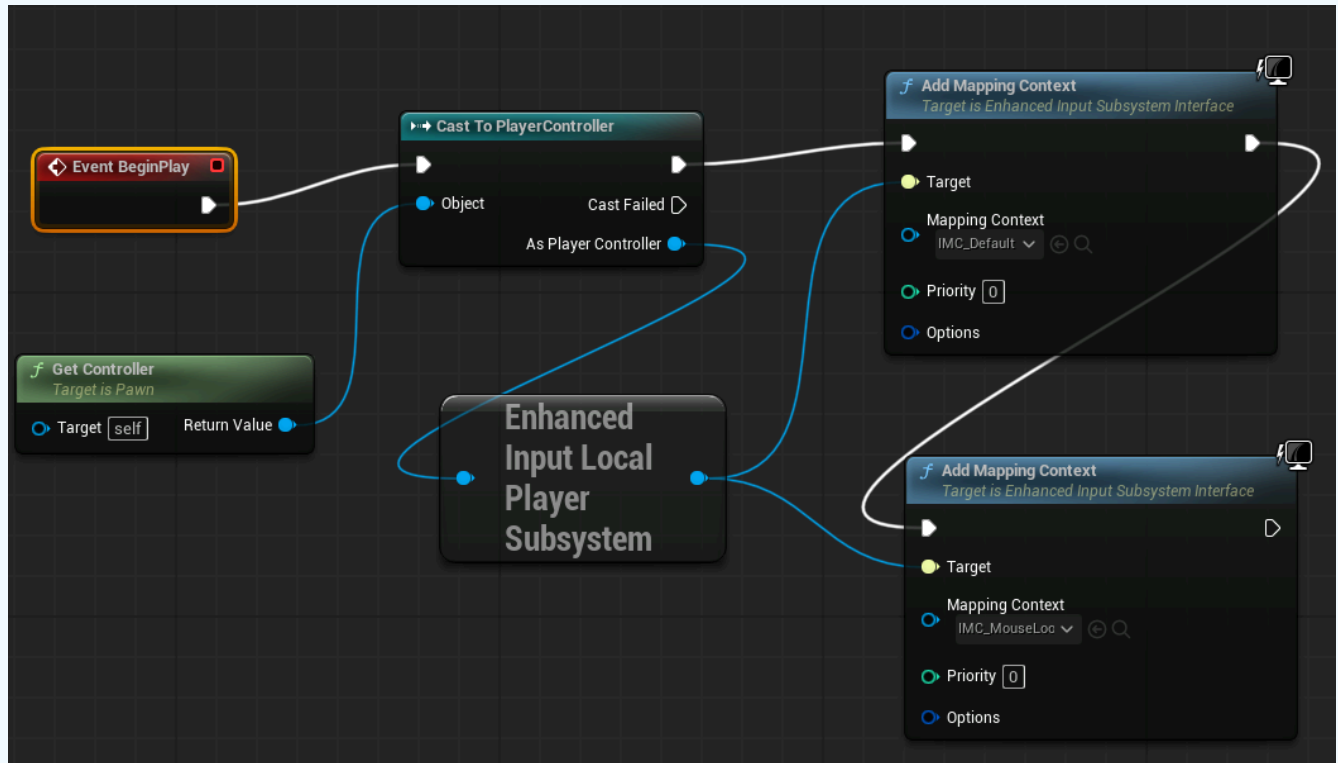
Per utilizzare la mappa di gioco di avvio

1. Nell'editor UE, apri la pagina Impostazioni del progetto, mappe e modalità ed espandi la sezione Mappe predefinite.
2. Per Editor Startup Map, seleziona StartupMap "" dall'elenco a discesa. Potrebbe essere necessario cercare il file, che si trova in... > Unreal Projects/[project-name]/Plugins/Amazon GameLift Servers Plugin Content/Maps.
3. Per Game Default Map, seleziona la stessa StartupMap "" dall'elenco a discesa.
4. Per Server Default Map, seleziona «Lv1_ThirdPerson» per Unreal Engine 5.6 o versioni successive o "" ThirdPersonMap per le versioni precedenti. Questa è una mappa predefinita inclusa nel progetto di gioco. Questa mappa è progettata per due giocatori.
5. Apri il pannello dei dettagli per la mappa predefinita del server. Imposta GameMode Override su «Nessuno».
6. Espandi la sezione Modalità predefinite e imposta la modalità di gioco Global Default Server sulla modalità di gioco che hai aggiornato per l'integrazione del server.

Dopo aver apportato queste modifiche al progetto, sei pronto per creare i componenti del gioco.

Note

Per Unreal Engine 5.6 o versioni successive, se non riesci a spostare il personaggio dopo esserti connesso al server di gioco, aggiorna il ThirdPersonCharacter progetto BP_ per aggiungere il contesto di mappatura degli input e come mostrato di seguito: IMC_Default IMC_MouseLook



Package dei componenti del gioco

Per impacchettare le build del server di gioco e del client di gioco

1. Apri il progetto di gioco in una versione originale dell'editor di Unreal Engine.
2. Se usi Unreal Engine 5.6 o versione successiva, vai a Modifica, Impostazioni del progetto, Pacchetto. Trova Cook everything nella directory dei contenuti del progetto e abilitalo.
3. Usa l'editor per impacchettare le build del client di gioco e del server.
 - a. Scegli un bersaglio. Vai a Piattaforme, Windows e seleziona una delle seguenti opzioni:
 - Server: [your-application-name]Server
 - Client: [your-application-name]Client

- b. Avvia la compilazione. Vai a Platform, Windows, Package Project.

Ogni processo di confezionamento genera un file eseguibile: `[your-application-name]Client.exe` o `[your-application-name]Server.exe`.

Nel plugin, imposta i percorsi degli eseguibili di compilazione del client e del server sulla tua workstation locale.

Plugin per Unreal: ospita il tuo gioco localmente con Anywhere Amazon GameLift Servers

Usa questo flusso di lavoro per configurare la tua workstation locale come host di server di gioco utilizzando una flotta Anywhere. Puoi utilizzarlo per testare l'integrazione del server di gioco prima di distribuirlo su una flotta gestita basata su cloud. Può anche essere utile per i test locali durante lo sviluppo iterativo di giochi.

Per avviare il flusso di lavoro Amazon GameLift Servers Anywhere:

- Nella barra degli strumenti principale dell'editor Unreal, scegli il Amazon GameLift Servers menu e seleziona Host with Anywhere. Questa azione apre la pagina del plugin Deploy Anywhere, che presenta un processo in sei fasi per integrare, creare e avviare i componenti del gioco.

Passaggio 1: imposta il tuo profilo

Scegli il profilo che desideri utilizzare quando segui questo flusso di lavoro. Il profilo selezionato influisce su tutte le fasi del flusso di lavoro. Tutte le risorse create sono associate all' AWS account del profilo e collocate nella AWS regione predefinita del profilo. Le autorizzazioni dell'utente del profilo determinano l'accesso alle AWS risorse e alle azioni.

Per impostare un profilo utente

1. Seleziona un profilo dall'elenco a discesa dei profili disponibili. Se non hai ancora un profilo o desideri crearne uno nuovo, vai al GameLift menu Amazon e scegli Imposta profili AWS utente.
2. Se lo stato di bootstrap non è «Attivo», scegli il profilo Bootstrap e attendi che lo stato passi a «Attivo».

Passaggio 2: configura il codice del gioco

In questo passaggio, prepara le build del server di gioco e del client di gioco su cui lavorare Amazon GameLift Servers. Se non hai ancora integrato il codice di gioco, consulta [Plugin per Unreal: integra il tuo codice di gioco](#). Inserisci i percorsi degli eseguibili di gioco sulla tua workstation locale.

- **Server di gioco:** integra il tuo server di gioco con l'SDK del server Amazon GameLift Servers e crea un pacchetto per la build del tuo server di gioco. Per istruzioni, consulta [Plugin per Unreal: integra il tuo codice di gioco](#). Il server di gioco deve essere integrato con l'SDK del server per stabilire una comunicazione con il Amazon GameLift Servers servizio e rispondere alle richieste di avvio di nuove sessioni di gioco e accettare connessioni con i client di gioco.
- **Client di gioco:** è necessario almeno un client di gioco in grado di connettersi al server di gioco utilizzando l'indirizzo IP e le informazioni sulla porta. Se non hai ancora configurato i componenti del client di gioco Amazon GameLift Servers, puoi utilizzare lo AWS CLI strumento per richiedere manualmente nuove sessioni di gioco, ottenere informazioni di connessione e utilizzare tali informazioni per connettere il client di gioco.

A un certo punto, avrai bisogno di un servizio di backend per inviare nuove richieste di sessioni di gioco al Amazon GameLift Servers servizio e inoltrare le informazioni di connessione a un client di gioco. Puoi utilizzare le mappe di test incluse nel plugin per aggiungere Amazon GameLift Servers funzionalità client al tuo progetto di gioco. Per assistenza nella creazione di una soluzione personalizzata, consulta [Integra la funzionalità del client di Amazon GameLift Servers gioco](#).

Fase 3: Connect a una flotta Anywhere

In questa fase, si designa una flotta Anywhere da utilizzare. Una flotta Anywhere definisce una raccolta di risorse di elaborazione, che possono essere posizionate ovunque, per l'hosting di server di gioco.

- Se l' AWS account che stai utilizzando attualmente dispone di flotte Anywhere esistenti, apri il campo a discesa Fleet name e scegli una flotta. Questo menu a discesa mostra solo le flotte Anywhere nella AWS regione per il profilo utente attualmente attivo.
- Se non ci sono flotte esistenti o desideri crearne una nuova, scegli Crea una nuova flotta Anywhere e fornisci un nome per la flotta.

Dopo aver scelto una flotta Anywhere per il tuo progetto, Amazon GameLift Servers verifica che lo stato della flotta sia attivo e visualizza l'ID della flotta. Puoi tenere traccia dello stato di avanzamento di questa richiesta nel registro di output dell'editor Unreal.

Fase 4: Registra la tua postazione di lavoro

In questo passaggio, registri la tua workstation locale come risorsa di elaborazione nella nuova flotta Anywhere.

Per registrare la workstation come elaborazione Anywhere

1. Inserisci un nome di calcolo per il tuo computer locale. Se aggiungi più di un computer nel parco dati, i nomi devono essere univoci.
2. Fornisci un indirizzo IP per il tuo computer locale. L'impostazione predefinita di questo campo è l'indirizzo IP pubblico della macchina. Puoi anche usare localhost (127.0.0.1) purché utilizzi il client e il server di gioco sulla stessa macchina.
3. Scegli Register compute. Puoi tenere traccia dello stato di avanzamento di questa richiesta nel registro di output dell'editor Unreal.

In risposta a questa azione, Amazon GameLift Servers verifica che sia in grado di connettersi al computer e restituisce informazioni sul calcolo appena registrato. Crea inoltre gli argomenti della console necessari agli eseguibili di gioco per inizializzare la comunicazione con il servizio. Amazon GameLift Servers

Passaggio 5: Generazione del token di autenticazione

I processi del server di gioco in esecuzione sul computer Anywhere richiedono un token di autenticazione per effettuare chiamate al Amazon GameLift Servers servizio. Il plug-in genera e memorizza automaticamente un token di autenticazione per la flotta Anywhere ogni volta che avvia il server di gioco dal plug-in. Il valore del token di autenticazione viene memorizzato come argomento della riga di comando, che il codice del server può recuperare in fase di esecuzione.

Gli esempi di codice sopra riportati consentono inoltre di utilizzare [AWS Signature Version 4 \(SigV4\)](#) per le richieste API. SigV4 è il protocollo di AWS firma per aggiungere informazioni di autenticazione alle richieste API.

Non è necessario intraprendere alcuna azione in questo passaggio.

Passaggio 6: Avvia il gioco

A questo punto, hai completato tutte le attività necessarie per avviare e giocare alla partita multiplayer su una workstation locale utilizzando Amazon GameLift Servers.

Per giocare al gioco ospitato

1. Avvia il tuo server di gioco. Il server di gioco ti avviserà Amazon GameLift Servers quando sarà pronto per ospitare le sessioni di gioco.
2. Avvia il client di gioco e usa la nuova funzionalità per iniziare una nuova sessione di gioco. Questa richiesta viene inviata Amazon GameLift Servers tramite il nuovo servizio di backend. In risposta Amazon GameLift Servers, chiama il server di gioco, in esecuzione sul computer locale, per avviare una nuova sessione di gioco. Quando la sessione di gioco è pronta per accettare giocatori, Amazon GameLift Servers fornisce le informazioni di connessione per consentire al client di gioco di partecipare alla sessione di gioco.

Plugin per Unreal: distribuisce il tuo gioco su una flotta EC2 gestita

In questo flusso di lavoro, distribuisce il gioco per l'hosting su risorse di elaborazione basate su cloud gestite da Amazon GameLift Servers. Carica la build del tuo server di gioco integrato nel servizio per la Amazon GameLift Servers distribuzione. Se non hai ancora integrato il codice di gioco, consulta [Plugin per Unreal: integra il tuo codice di gioco](#). Una volta completato questo flusso di lavoro, avrai un client di gioco funzionante in grado di connettersi ai tuoi server di gioco nel cloud.

Per avviare il flusso di Amazon GameLift Servers lavoro gestito di Amazon EC2:

- Nella barra degli strumenti principale dell'editor Unreal, scegli il Amazon GameLift Servers menu e seleziona Host with Managed EC2. Questa azione apre la pagina del plug-in Deploy Amazon EC2 Fleet, che presenta un processo in sei fasi per integrare, creare, distribuire e avviare i componenti del gioco.

Passaggio 1: imposta il tuo profilo

Scegli il profilo che desideri utilizzare quando segui questo flusso di lavoro. Il profilo selezionato influisce su tutte le fasi del flusso di lavoro. Tutte le risorse create sono associate all' AWS account del profilo e collocate nella AWS regione predefinita del profilo. Le autorizzazioni dell'utente del profilo determinano l'accesso alle AWS risorse e alle azioni.

Per impostare un profilo utente

1. Seleziona un profilo dall'elenco a discesa dei profili disponibili. Se non hai ancora un profilo o desideri crearne uno nuovo, vai al GameLift menu Amazon e scegli Imposta profili AWS utente.
2. Se lo stato di bootstrap non è «Attivo», scegli il profilo Bootstrap e attendi che lo stato passi a «Attivo».

Passaggio 2: configura il codice del gioco

In questo passaggio, prepara le build del server di gioco e del client di gioco per utilizzarle con l'SDK del server Amazon GameLift Servers C++ per Unreal. Se non hai ancora integrato il codice di gioco e non hai ancora creato gli eseguibili per client e server di gioco, vedi. [Plugin per Unreal: integra il tuo codice di gioco](#) Inserisci i percorsi degli eseguibili di gioco sulla tua workstation locale.

In questa fase del flusso di lavoro, il plugin fornisce collegamenti alle istruzioni e al codice sorgente per configurare una versione sorgente di Unreal Editor. È necessario utilizzare la versione generata dal codice sorgente quando si creano i componenti client e server.

Dopo aver creato un server di gioco integrato con l'SDK del server, completa le seguenti attività per prepararlo al caricamento per l'hosting. Amazon GameLift Servers

Per preparare la build del server per la distribuzione su cloud (Windows)

Nella `WindowsServer` cartella in cui l'editor Unreal archivia i file di build del server per impostazione predefinita, apporta le seguenti aggiunte:

1. Copia lo script di installazione della build del server nella radice della **WindowsServer** cartella. Lo script di installazione è incluso nel download del plugin. Cerca il file `[project-name]/Plugins/Resources/CloudFormation/extra_server_resources/install.bat`. Amazon GameLift Servers utilizza questo file per installare la build del server sui tuoi computer di hosting.
2. Copia il **VC_redist.x64.exe** file nella radice della **WindowsServer** cartella. Puoi saltare questo passaggio se utilizzi Unreal Engine versione 5.6 o successiva. Questo file è incluso nell'installazione di Visual Studio. Di solito si trova in `C:/Program Files (x86)/Microsoft Visual Studio/2019/Professional/VC/Redist/MSVC/v142`.
3. Aggiungi i file della libreria OpenSSL alla build del tuo server di gioco. . Puoi saltare questo passaggio se il tuo server di gioco è integrato con il server SDK 5.3 o successivo. Questa versione è inclusa nel Amazon GameLift Servers plugin per Unreal, versione 3.0 o successiva.

Individua e copia manualmente le librerie OpenSSL nella directory del pacchetto di build del gioco all'indirizzo. <YourGame>/Binaries/Win64 È necessario utilizzare la stessa versione di OpenSSL utilizzata dalla versione di Unreal Engine 5. Le build di gioco distribuite con le librerie OpenSSL sbagliate non saranno in grado di comunicare con il servizio. Amazon GameLift Servers

Cerca le librerie OpenSSL nella fonte del tuo motore di gioco. La posizione varia a seconda dell'ambiente di sviluppo:

In Windows:

- [ENGINE_ROOT_DIR]\Engine\Extras\ThirdPartyNotUE\libmobiledevice\x64\libssl-1_1-x64.dll
- [ENGINE_ROOT_DIR]\Engine\Extras\ThirdPartyNotUE\libmobiledevice\x64\libcrypto-1_1-x64.dll

In Linux:

- Engine/Source/Thirdparty/OpenSSL/1.1.1n/include/libssl.so.1.1
- Engine/Source/Thirdparty/OpenSSL/1.1.1n/include/libcrypto.so.1.1

Per preparare la build del server per l'implementazione su cloud (Linux)

Per istruzioni più dettagliate sulla preparazione di un server di gioco creato per Linux, consulta [Creazione dell'SDK del server Amazon GameLift Servers per Unreal Engine 5 su Amazon Linux](#).

1. Designate una directory di lavoro per organizzare i file di build. La struttura della directory di lavoro viene distribuita così com'è su ogni computer di hosting. Aggiungi il tuo server di gioco basato su Linux e tutti i file dipendenti.
2. Crea uno script di installazione della build del server nella radice della tua directory di lavoro. Se necessario, crea un `install.sh` file e aggiungi tutti i comandi necessari per installare correttamente la build del server di gioco. Amazon GameLift Servers utilizza questo file per installare la build del server su ogni risorsa di hosting EC2.
3. Aggiungi i file della libreria OpenSSL alla build del tuo server di gioco. Puoi saltare questo passaggio se il tuo server di gioco è integrato con il server SDK 5.3 o successivo.

Individua e copia manualmente le librerie. È necessario utilizzare la stessa versione di OpenSSL utilizzata dalla versione di Unreal Engine 5. Le build di gioco distribuite con le librerie OpenSSL sbagliate non saranno in grado di comunicare con il servizio. Amazon GameLift Servers

- a. Cerca le librerie OpenSSL nella fonte del tuo motore di gioco. La posizione varia a seconda dell'ambiente di sviluppo:

In Windows:

- `[ENGINE_ROOT_DIR]\Engine\Extras\ThirdPartyNotUE\libmobiledevice\x64\libssl-1_1-x64.dll`
- `[ENGINE_ROOT_DIR]\Engine\Extras\ThirdPartyNotUE\libmobiledevice\x64\libcrypto-1_1-x64.dll`

In Linux:

- `Engine/Source/Thirdparty/OpenSSL/1.1.1n/include/libssl.so.1.1`
 - `Engine/Source/Thirdparty/OpenSSL/1.1.1n/include/libcrypto.so.1.1`
- b. Quando trovi le librerie OpenSSL, copiale nella directory del pacchetto di build del gioco all'indirizzo. `<YourGame>/Binaries/Linux`

Passaggio 3: Seleziona lo scenario di implementazione

In questo passaggio, scegli la soluzione di hosting del gioco che desideri implementare in questo momento. Puoi avere più distribuzioni del gioco, utilizzando uno qualsiasi degli scenari.

- Flotta a regione singola: distribuisce il server di gioco su un'unica flotta di risorse di hosting nella regione predefinita del profilo attivo. AWS Questo scenario è un buon punto di partenza per testare l'integrazione del server AWS e la configurazione della build del server. Implementa le seguenti risorse:
 - AWS fleet (On-Demand) con la build del server di gioco installata e funzionante.
 - Pool di utenti e client Amazon Cognito per consentire ai giocatori di autenticarsi e iniziare una partita.
 - Autorizzatore di gateway API che collega il pool di utenti con. APIs
 - Web ACI per limitare le chiamate eccessive dei giocatori al gateway API.

- API gateway + funzione Lambda per consentire ai giocatori di richiedere uno slot di gioco. Questa funzione chiama `CreateGameSession()` se non ce ne sono disponibili.
- API gateway + funzione Lambda per consentire ai giocatori di ottenere informazioni di connessione per la loro richiesta di gioco.
- FlexMatch flotta: distribuisce il tuo server di gioco su una serie di flotte e configura un FlexMatch matchmaker con le regole per creare partite tra giocatori. Questo scenario utilizza un hosting Spot a basso costo con una struttura multi-flotta e più sedi per una disponibilità duratura. Questo approccio è utile quando sei pronto per iniziare a progettare un componente matchmaker per la tua soluzione di hosting. In questo scenario, creerai le risorse di base per questa soluzione, che potrai personalizzare in seguito, se necessario. Implementa le seguenti risorse:
 - FlexMatch configurazione del matchmaking e regole di matchmaking impostate per accettare le richieste dei giocatori e formare partite.
 - Tre AWS flotte con la configurazione del server di gioco installata e funzionante in più località. Include due flotte Spot e una flotta On-Demand come backup.
 - AWS coda per il posizionamento delle sessioni di gioco che soddisfa le richieste di partite proposte trovando la migliore risorsa di hosting possibile (in base a fattibilità, costo, latenza dei giocatori, ecc.) e avviando una sessione di gioco.
 - Pool di utenti e client Amazon Cognito per consentire ai giocatori di autenticarsi e iniziare una partita.
 - Autorizzatore di gateway API che collega il pool di utenti con. APIs
 - Web ACL per limitare le chiamate eccessive dei giocatori al gateway API.
 - API gateway + funzione Lambda per consentire ai giocatori di richiedere uno slot di gioco. Questa funzione chiama `StartMatchmaking()`.
 - API gateway + funzione Lambda per consentire ai giocatori di ottenere informazioni di connessione per la loro richiesta di gioco.
 - Tabelle Amazon DynamoDB per archiviare i ticket di matchmaking per i giocatori e le informazioni sulle sessioni di gioco.
 - Argomento SNS + funzione Lambda per `GameSessionQueue` gestire gli eventi.

Passaggio 4: imposta i parametri di gioco

In questo passaggio, descrivi il gioco su cui caricarlo AWS;

- Nome della build del server: fornisci un nome significativo per la build del server di gioco. AWS usa questo nome per fare riferimento alla copia della build del server che viene caricata e utilizzata per le distribuzioni.
- Sistema operativo basato su server: inserisci il sistema operativo su cui è costruito il server. Indica il AWS tipo di risorse di calcolo da utilizzare per ospitare il gioco.
- Cartella del server di gioco: identifica il percorso della cartella di build del server locale.
- Build del server di gioco: identifica il percorso dell'eseguibile del server di gioco.
- Percorso del client di gioco: identifica il percorso dell'eseguibile del client di gioco.
- Output della configurazione del client: questo campo deve puntare a una cartella nella build del client che contiene la AWS configurazione. Cercalo nella seguente posizione:[client-build]/[project-name]/Content/CloudFormation.

Fase 5: Implementazione dello scenario

In questo passaggio, distribuisce il gioco su una soluzione di hosting cloud in base allo scenario di distribuzione scelto. Questo processo può richiedere diversi minuti per AWS la convalida della build del server, il provisioning delle risorse di hosting, l'installazione del server di gioco, l'avvio dei processi del server e la preparazione per ospitare sessioni di gioco.

Per iniziare la distribuzione, scegli Deploy. CloudFormation Puoi monitorare lo stato del tuo hosting di giochi qui. Per informazioni più dettagliate, puoi accedere alla console di AWS gestione AWS e visualizzare le notifiche degli eventi. Assicurati di accedere utilizzando lo stesso account, utente e AWS regione del profilo utente attivo nel plug-in.

Una volta completata la distribuzione, il server di gioco è installato su un'istanza AWS EC2. Almeno un processo server è in esecuzione ed è pronto per iniziare una sessione di gioco.

Passaggio 6: Avvia il client

A questo punto, hai completato tutte le attività necessarie per avviare e giocare al gioco multiplayer ospitato con Amazon GameLift Servers. Per giocare, avvia un'istanza del tuo client di gioco.

Se hai implementato lo scenario a flotta singola, puoi aprire una singola istanza client con un giocatore, accedere alla mappa del server e spostarti. Apri istanze aggiuntive del client di gioco per aggiungere un secondo giocatore alla stessa mappa di gioco del server.

Se hai implementato FlexMatch lo scenario, la soluzione attende che almeno due client vengano messi in coda per il posizionamento della sessione di gioco prima che i giocatori possano accedere alla mappa del server.

Plugin per Unreal: distribuisce il gioco su una flotta di container gestita

Usa questo flusso di lavoro guidato dai plug-in per creare un'immagine del contenitore per il tuo server di gioco e distribuirlo su una soluzione di hosting basata su container. Se non hai ancora integrato il codice di gioco, consulta [Plugin per Unreal: integra il tuo codice di gioco](#). Una volta completato con successo questo flusso di lavoro, il server di gioco containerizzato è in esecuzione nel cloud e puoi utilizzare il plug-in per avviare un client di gioco, connetterti a una sessione di gioco e giocare.

Prima di iniziare

Questo flusso di lavoro presuppone che tu abbia completato le seguenti attività.

- Integra il codice del server di gioco con l'SDK Amazon GameLift Servers del server. Il server di gioco ospitato deve essere in grado di comunicare con il Amazon GameLift Servers servizio in modo da poter rispondere alle richieste di avvio di nuove sessioni di gioco e segnalare lo stato della sessione di gioco. Se non hai completato questa operazione, ti consigliamo di seguire prima il flusso di lavoro del plug-in Host with Anywhere. Per istruzioni sulla preparazione del codice del server di gioco, consulta [Aggiorna il codice del server di gioco](#). Per una flotta di container gestita, devi integrare il gioco con la versione 5.2 o successiva dell'SDK del server.

Note

Se utilizzi la mappa di gioco di avvio, questa operazione è già stata eseguita per te.

- Package dell'eseguibile del server di gioco per eseguirlo su Linux. [Se stai sviluppando su Windows e stai integrando C++ Server SDK versione 5.2.x o precedente, dovrai utilizzare l'Unreal cross compile toolkit](#). In alternativa, puoi configurare uno spazio di lavoro Linux separato o utilizzare uno strumento come il sottosistema Windows per Linux (WSL).
- Raccogli i file da distribuire con la build del tuo server di gioco. Sul computer locale, crea una directory di lavoro per organizzare i file, che verrà incorporata nell'immagine del contenitore del server di gioco. Questi potrebbero includere dipendenze di gioco, uno script per avviare i server di gioco e altri processi all'avvio di un contenitore, ecc.
- Aggiungi i file della libreria OpenSSL per la build del tuo server di gioco. Puoi saltare questo passaggio se il tuo server di gioco è integrato con il server SDK 5.3 o successivo.

Individua e copia manualmente le librerie OpenSSL nella directory del pacchetto di build del gioco all'indirizzo. <YourGame>/Binaries/Win64 È necessario utilizzare la stessa versione di OpenSSL utilizzata dalla versione di Unreal Engine 5. Le build di gioco distribuite con le librerie OpenSSL sbagliate non saranno in grado di comunicare con il servizio. Amazon GameLift Servers

Per trovare le librerie OpenSSL, cerca nella fonte del tuo motore di gioco. La posizione varia a seconda dell'ambiente di sviluppo:

In Windows:

- [ENGINE_ROOT_DIR]\Engine\Extras\ThirdPartyNotUE\libmobiledevice\x64\libssl-1_1-x64.dll
- [ENGINE_ROOT_DIR]\Engine\Extras\ThirdPartyNotUE\libmobiledevice\x64\libcrypto-1_1-x64.dll

In Linux:

- Engine/Source/Thirdparty/OpenSSL/1.1.1n/include/libssl.so.1.1
- Engine/Source/Thirdparty/OpenSSL/1.1.1n/include/libcrypto.so.1.1
- Integra il codice del client di gioco con Amazon GameLift Servers. Un modo per completare questa operazione consiste nell'aggiungere una risorsa di esempio (inclusa nel plug-in) già integrata. Per indicazioni sulla preparazione del codice client di gioco, consulta [Integra la mappa di gioco del tuo client](#).
- Installa Docker sul tuo computer locale. È necessario che questo strumento sia installato se desideri che il plug-in crei immagini dei contenitori per te e le invii a un repository ECR. In alternativa, puoi eseguire queste attività manualmente e indicare al plug-in di utilizzare un'immagine del contenitore esistente. Per ulteriori informazioni sulla creazione manuale dell'immagine, consulta [Creare un'immagine del contenitore per Amazon GameLift Servers](#).

Per avviare il flusso di lavoro dei contenitori Amazon GameLift Servers gestiti:

- Nella barra degli strumenti principale dell'editor Unreal, scegli il Amazon GameLift Servers menu e seleziona Contenitori gestiti. Questa azione apre la pagina del plug-in Host with Managed Containers, che presenta un step-by-step processo per creare un'immagine del contenitore con la build del server di gioco, distribuirla in una flotta di container e avviare il gioco.

Passaggio 0: imposta il tuo profilo

Questa sezione mostra il profilo utente attualmente selezionato. Verifica che il profilo utente corrente sia quello che desideri utilizzare per questo flusso di lavoro. Tutte le risorse create in questo flusso di lavoro sono associate all' AWS account del profilo e collocate nella AWS regione predefinita del profilo. Le autorizzazioni dell'utente del profilo determinano l'accesso alle AWS risorse e alle azioni.

Potrebbe essere necessario modificare il profilo utente selezionato se:

- Al momento non è selezionato alcun profilo.
- Vuoi selezionare un profilo diverso o creare un nuovo profilo.
- È necessario avviare il profilo selezionato (se lo stato di bootstrap è inattivo).

Per impostare o modificare il profilo utente selezionato

- Nel Amazon GameLift Servers menu, scegli Open AWS Access Credentials.

Fase 1: Valutare la disponibilità del contenitore

Prima di distribuire il tuo server di gioco in una flotta di container, devi impacchettarlo in un'immagine del contenitore e archivarlo in un repository Amazon ECR. Il plug-in può completare queste attività per te oppure puoi eseguirle manualmente. In questo passaggio, fornisci informazioni sullo stato dell'immagine del contenitore e dell'archivio ECR.

Usa le domande di valutazione per indicare al plugin quali passi deve compiere:

- Crea una nuova immagine del contenitore. Se scegli questa opzione, nel passaggio successivo ti verrà richiesta la posizione della directory di build del server di gioco e l'eseguibile della build. Il plugin utilizza un modello Dockerfile (fornito da Amazon GameLift Servers) e lo configura automaticamente per il gioco. Puoi visualizzare il modello all'indirizzo. [Crea un'immagine del contenitore per Amazon GameLift Servers](#) Dopo aver scelto questa opzione, indica dove desideri che il plugin memorizzi la nuova immagine:
- Crea un nuovo repository Amazon ECR e inviaci l'immagine del contenitore. Il plug-in crea un repository ECR privato utilizzando l' AWS account e lo utilizza come impostazione predefinita Regione AWS nel profilo utente selezionato.
- Invia l'immagine del contenitore a un repository Amazon ECR creato in precedenza. Se scegli questa opzione, il passaggio successivo ti chiederà di selezionare un repository Amazon ECR esistente da un elenco. L'elenco include tutti i repository Amazon ECR per l' AWS account e

quelli predefiniti Regione AWS nel profilo utente selezionato. Puoi selezionare un repository pubblico o privato.

- Usa un'immagine del contenitore esistente. Se hai creato un'immagine manualmente, ti consigliamo di utilizzare il modello Dockerfile fornito da Amazon GameLift Servers, disponibile all'indirizzo. [Crea un'immagine del contenitore per Amazon GameLift Servers](#) Dopo aver scelto questa opzione, indica dove si trova l'immagine.
- Un'immagine generata da Docker archiviata localmente. Se scegli questa opzione, il plug-in crea un nuovo repository privato Amazon ECR e vi invia il file di immagine locale. Il passaggio successivo richiederà un ID immagine, che il plug-in utilizza per localizzare il file di immagine.
- Un'immagine del contenitore già archiviata in un repository Amazon ECR. Se scegli questa opzione, il passaggio successivo ti chiederà di selezionare un repository e un'immagine Amazon ECR esistenti da un elenco. L'elenco include tutti i repository Amazon ECR per l'AWS account e quelli predefiniti Regione AWS nel profilo utente selezionato. Puoi selezionare un repository pubblico o privato.

Fase 2: Configurare la distribuzione delle immagini

In questo passaggio, fornisci le informazioni necessarie al plug-in per distribuire l'immagine del container in una flotta di container. Questo passaggio richiede le seguenti informazioni:

- La posizione della build del server di gioco, dell'immagine del contenitore o del repository Amazon ECR, in base alle selezioni effettuate nella Fase 1.
- Lo scenario da utilizzare per la distribuzione dei container gestiti.
- Il percorso di output della configurazione del client. Seleziona la cartella nella build del client che contiene la tua AWS configurazione. Cercalo nella seguente posizione: `[client-build]/[project-name]/Content/CloudFormation`.
- Impostazioni di distribuzione opzionali. Questa sezione contiene le impostazioni di configurazione che il plugin utilizza per impostazione predefinita. È possibile modificarli o mantenere i valori predefiniti
 - Per impostazione predefinita, il nome del gioco è impostato sul nome del progetto di gioco. Tutte le AWS risorse create dal plugin fanno riferimento al valore del nome del gioco.
 - L'intervallo di porte, il limite di memoria e il limite di vCPU sono impostazioni di configurazione per la flotta di container. Per ulteriori informazioni sulla personalizzazione di questi valori, consulta [Configurare le connessioni di rete](#) l'intervallo delle porte di connessione e [Imposta i limiti delle risorse](#) i limiti delle risorse.

- Il tag dell'immagine del contenitore viene utilizzato per classificare le immagini del contenitore in Amazon ECR. Il valore predefinito è `unreal-gamelift-plugin`.
- Nome del repository Amazon ECR. Puoi modificare questo campo per suggerire un nome personalizzato solo quando il plug-in sta creando un repository ECR per te. Il valore predefinito è `unreal-game lift-plugin-ecr-repository`.

Opzioni dello scenario di implementazione

Flotta di container in un'unica regione

Questo scenario distribuisce il tuo server di gioco su una singola flotta di container. È un buon punto di partenza per testare l'integrazione del server AWS e la configurazione del container. Implementa le seguenti risorse.

- Amazon GameLift Servers La definizione del gruppo di container descrive come distribuire ed eseguire le immagini dei container su una flotta di container.
- Amazon GameLift Servers flotta di container (On-Demand) con il container del server di gioco installato e funzionante, con alias.
- Pool di utenti e client Amazon Cognito per consentire ai giocatori di autenticarsi e iniziare una partita.
- Autorizzatore API Gateway che collega il pool di utenti con APIs.
- Elenco di controllo degli accessi Web (ACL) per limitare le chiamate eccessive dei giocatori all'API Gateway.
- Servizio di backend per effettuare richieste al Amazon GameLift Servers servizio per conto dei client di gioco, ad esempio per richiedere sessioni di gioco e partecipare ai giochi:
 - API Gateway + funzione Lambda per consentire ai giocatori di richiedere uno slot per la sessione di gioco. Questa funzione chiama `CreateGameSession()` se non sono disponibili slot aperti.
 - API Gateway + funzione Lambda per consentire ai giocatori di ottenere informazioni di connessione per la loro richiesta di gioco.

Flotta di container a regione singola con FlexMatch

Questo scenario distribuisce il server di gioco su una flotta di container, configura il posizionamento delle sessioni di gioco, e imposta il matchmaking. FlexMatch Questo scenario è utile quando sei pronto per iniziare a progettare un matchmaker personalizzato per la tua soluzione di hosting. Utilizza

questo scenario per creare le risorse di base per questa soluzione, che potrai personalizzare in seguito, se necessario. Implementa le seguenti risorse:

- Amazon GameLift Serversdefinizione del gruppo di container che descrive come distribuire ed eseguire le immagini dei container su una flotta di container.
- Amazon GameLift Serversflotta di container (On-Demand) con il container del server di gioco installato e funzionante, con alias.
- FlexMatchconfigurazione del matchmaking e regole di matchmaking impostate per accettare le richieste dei giocatori e formare partite.
- Amazon GameLift Serverscoda di sessione di gioco che soddisfa le richieste di partite proposte trovando la migliore risorsa di hosting possibile (in base a fattibilità, costo, latenza dei giocatori, ecc.) e avviando una sessione di gioco.
- Pool di utenti e client Amazon Cognito per consentire ai giocatori di autenticarsi e iniziare una partita.
- Autorizzatore API Gateway che collega il pool di utenti con APIs.
- Elenco di controllo degli accessi Web (ACL) per limitare le chiamate eccessive dei giocatori all'API Gateway.
- Servizio di backend per effettuare richieste al Amazon GameLift Servers servizio per conto dei client di gioco, ad esempio per richiedere sessioni di gioco e partecipare ai giochi:
 - API Gateway + funzione Lambda per consentire ai giocatori di richiedere uno slot per la sessione di gioco. Questa funzione chiama `StartMatchmaking()` se non sono disponibili slot aperti.
 - API Gateway + funzione Lambda per consentire ai giocatori di ottenere informazioni di connessione per la loro richiesta di gioco.
- Tabelle DynamoDB per archiviare i ticket di matchmaking per i giocatori e le informazioni sulle sessioni di gioco.
- Argomento Amazon SNS + funzione Lambda per gestire gli eventi. `GameSessionQueue`

Implementa una flotta di container

Quando la configurazione della flotta è completa, scegli il pulsante Implementa la flotta di container per avviare l'implementazione. Questo processo può richiedere diversi minuti mentre il plug-in crea un'immagine del container e la invia a ECR, fornisce le risorse di hosting per la flotta di container, distribuisce la flotta e altre AWS risorse per lo scenario della soluzione di hosting selezionata.

Quando si avvia una distribuzione, è possibile monitorare lo stato di avanzamento di ogni fase. A seconda della configurazione, i passaggi potrebbero includere quanto segue:

- Configurazione dell'immagine del contenitore
- Creazione di un repository Amazon ECR
- Creare un'immagine e passare ad Amazon ECR
- Creazione della definizione di un gruppo di contenitori
- Creazione di una flotta di container

Per informazioni più dettagliate sulla distribuzione, scegli Visualizza nella console AWS di gestione. Quando la flotta di container raggiunge lo stato attivo, utilizza attivamente container con processi server pronti per ospitare sessioni di gioco.

Una volta completata l'implementazione, disponi di una flotta di container funzionante pronta per ospitare sessioni di gioco e accettare connessioni tra giocatori.

Non puoi interrompere una distribuzione in corso. Se la distribuzione entra in uno stato errato o fallisce, puoi ricominciare da capo utilizzando l'opzione Reset deployment.

Avvia il client

A questo punto, hai completato tutte le attività per avviare e giocare al gioco multiplayer ospitato con Amazon GameLift Servers. Per giocare, scegli Start Client per avviare un'istanza locale del tuo client di gioco.

- Se hai utilizzato lo scenario a flotta singola, apri un'istanza del client di gioco con un solo giocatore e accedi alla mappa del server per muoverti. Puoi aprire una seconda istanza del client di gioco per aggiungere un secondo giocatore alla stessa mappa di gioco del server.
- Se hai implementato FlexMatch lo scenario, la soluzione di hosting attende che almeno due client di gioco effettuino richieste di matchmaking. Apri almeno due istanze del tuo client di gioco con un giocatore. I due giocatori verranno abbinati e verrà richiesto di partecipare a una sessione di gioco per la partita.

Aggiorna una flotta di container

Se hai implementato con successo una soluzione di hosting di container gestiti, puoi utilizzare la funzionalità Update deployment. Questa opzione consente di aggiornare le impostazioni di configurazione per una flotta di container distribuita, senza dover creare una nuova flotta.

Quando aggiorni una distribuzione, puoi distribuire un'immagine del contenitore con una build diversa del server di gioco, modificare il repository Amazon ECR, scegliere uno scenario di distribuzione diverso e personalizzare le impostazioni di configurazione opzionali.

Quando sei pronto per distribuire le modifiche, scegli **Aggiorna**. Il tempo necessario per un aggiornamento della distribuzione è simile a quello di una distribuzione completa. Per informazioni dettagliate sulla distribuzione, scegli **Visualizza** nella console AWS di gestione.

Pulisci le risorse distribuite

Come best practice, pulisci AWS le risorse per la tua soluzione di container gestiti non appena non ne hai più bisogno. Potresti continuare a sostenere dei costi per queste risorse se non le rimuovi.

Elimina le seguenti risorse:

- **Stack di risorse del contenitore gestito.** Le risorse in questo stack dipendono dallo scenario di distribuzione selezionato. Per eliminare l'intero stack, usa la CloudFormation console. Gli stack generati dal Amazon GameLift Servers plugin utilizzano la seguente convenzione di denominazione: `GameLiftPluginForUnreal-{GameName}-Containers`. Attendi il completamento del processo di eliminazione dello stack prima di avviare una nuova distribuzione di contenitori gestiti nel plug-in. Per ulteriori informazioni, consulta [Eliminare uno stack dalla console CloudFormation](#).
- **Repository Amazon ECR.** Se hai utilizzato il plug-in per creare un repository per l'immagine del contenitore, potresti voler eliminare tutti i repository che non sono più necessari. Non è necessario eliminare un repository prima di reimpostare la distribuzione di contenitori gestiti. Se aggiorni o ripristini una distribuzione, il plug-in utilizzerà automaticamente lo stesso repository a meno che non venga richiesto di utilizzarne un altro. Per ulteriori informazioni, consulta [Eliminazione di un repository privato in Amazon ECR](#).

Amazon GameLift Serversplugin per Unity (server SDK 5.x)

Questo plugin aggiunge l'SDK e gli strumenti del server Amazon GameLift Servers C# all'editor Unity. Utilizza i flussi di lavoro guidati dell'interfaccia utente per integrare la funzionalità SDK del server nel tuo progetto di gioco e implementare una soluzione di Amazon GameLift Servers hosting per il tuo server di gioco.

Con il plug-in, puoi creare una soluzione di hosting funzionante di base e quindi ottimizzarla e personalizzarla secondo necessità. Configura una flotta Amazon GameLift Servers Anywhere con la tua workstation locale come host. Per l'hosting su cloud con EC2 gestito o flotte di container gestite,

implementa il tuo server di gioco con una soluzione completa per gestire le richieste di sessioni di gioco e le connessioni dei client.

Argomenti

- [Installa il plugin per il tuo progetto di gioco Unity](#)
- [Plugin per Unity: configura un profilo AWS utente](#)
- [Plugin per Unity: configura i test locali con Anywhere Amazon GameLift Servers](#)
- [Plugin per Unity: distribuisce il tuo gioco su una flotta EC2 gestita](#)
- [Plugin per Unity: distribuisce il gioco su una flotta di container gestita](#)

Installa il plugin per il tuo progetto di gioco Unity

[Scarica il Amazon GameLift Servers plugin per Unity da GitHub](#)

Consulta il readme del GitHub repository per informazioni su come installare il plugin per un progetto di gioco.

Il plugin include questi componenti:

- Moduli plug-in per l'editor Unity. Quando il plugin è installato, una nuova voce del menu principale consente di accedere alle Amazon GameLift Servers funzionalità.
- Librerie C# per l'API Amazon GameLift Servers di servizio con funzionalità lato client.
- Librerie C# per l'SDK del Amazon GameLift Servers server (versione 5.x).
- Esempi di contenuti di gioco, tra cui risorse e scene, per poterli provare Amazon GameLift Servers anche se non disponi di un gioco multigiocatore pronto per l'installazione.
- Configurazioni delle soluzioni, fornite come CloudFormation modelli, utilizzate dal plug-in per distribuire il server di gioco sul cloud per l'hosting.

Questo plugin utilizza AWS CloudFormation modelli per implementare soluzioni di hosting per scenari di gioco comuni. Utilizza queste soluzioni così come fornite o personalizzate secondo necessità per i tuoi giochi.

Plugin per Unity: configura un profilo AWS utente

Dopo aver installato il plugin, configura un profilo utente con un AWS account valido. Puoi mantenere più profili nel plugin, ma puoi selezionare solo un profilo alla volta. Ogni volta che lavori nel

plugin, seleziona un profilo da usare. Ogni pagina del flusso di lavoro mostra il profilo attualmente selezionato.

Il mantenimento di più profili consente di passare da una distribuzione di hosting all'altra. Ad esempio, è possibile configurare profili che utilizzano lo stesso AWS account ma vengono distribuiti in regioni diverse. AWS Oppure puoi configurare profili con AWS account o utenti e set di autorizzazioni diversi.

Note

Se hai installato la AWS CLI sulla tua workstation e hai già un profilo configurato, il Amazon GameLift Servers plugin lo rileverà e lo elencherà come profilo esistente. Il plugin seleziona automaticamente qualsiasi profilo denominato. [default] È possibile utilizzare un profilo esistente o crearne uno nuovo.

Tutti i profili devono essere avviati per configurare alcune AWS risorse necessarie nell'account utente.

Per gestire i tuoi profili AWS

1. Nella barra degli strumenti principale di Unity, scegli il Amazon GameLift Servers menu e seleziona Credenziali di AWS accesso. Questa azione apre il Amazon GameLift Servers plugin alla pagina Configura il tuo profilo utente.
2. Utilizza i pulsanti per creare un nuovo AWS account o configurare un profilo utente per un AWS account che già possiedi.
3. Se non disponi già di un profilo utente, ti verrà richiesto di inserire i dettagli del profilo e creare un nuovo profilo. Inserisci le informazioni che seguono:
 - Un AWS account. Se crei un nuovo AWS account, usa il link Console di gestione AWS e segui le istruzioni. Vedi [Creare un AWS account](#) per maggiori dettagli.
 - Un AWS utente con autorizzazioni all'uso Amazon GameLift Servers e altri AWS servizi richiesti. Consulta [Configurare un account AWS utente](#) le istruzioni sulla configurazione di un utente AWS Identity and Access Management (IAM) con Amazon GameLift Servers autorizzazioni e accesso programmatico con credenziali a lungo termine.
 - Credenziali per il tuo utente. AWS Queste credenziali sono costituite da un ID della chiave di AWS accesso e da una chiave AWS segreta. Per maggiori dettagli, consulta [Ottenere le chiavi di accesso](#).

- AWS regione. Questa è una posizione geografica in cui desideri creare AWS le tue risorse per l'hosting. Durante lo sviluppo, ti consigliamo di utilizzare un'area vicina alla tua posizione fisica. Scegli una regione dall'elenco delle [AWS regioni supportate](#).
4. Se il plugin rileva profili esistenti, visualizza un elenco di profili disponibili. Seleziona un profilo esistente dall'elenco o scegli Aggiungi un altro profilo per crearne uno nuovo.

Avvia un profilo utente

Tutti i profili devono essere avviati per essere utilizzati con il plugin. Amazon GameLift Servers Il bootstrap crea un bucket Amazon S3 specifico per il profilo. Viene utilizzato per archiviare configurazioni di progetto, creare artefatti e altre dipendenze. I bucket non vengono condivisi tra altri profili.

Il bootstrap implica la creazione di nuove AWS risorse e potrebbe comportare dei costi.

Per avviare il tuo profilo:

1. Nella pagina Credenziali di AWS accesso, controlla lo stato di bootstrap del profilo utente che desideri utilizzare. Se lo stato di bootstrap del profilo è «Inattivo» e non è presente alcun bucket S3 nell'elenco, devi avviare il profilo.
2. Seleziona il profilo che desideri utilizzare e scegli il profilo Bootstrap.
3. Attendi che lo stato del bootstrap passi a «Attivo». Ciò può richiedere alcuni minuti.

Plugin per Unity: configura i test locali con Anywhere Amazon GameLift Servers

In questo flusso di lavoro, aggiungete il codice di gioco del client e del server per verificarne la Amazon GameLift Servers funzionalità e utilizzate il plug-in per designare la workstation locale come host del server di gioco di prova. Una volta completate le attività di integrazione, utilizzate il plug-in per creare i componenti del client e del server di gioco.

Per avviare il flusso di lavoro Amazon GameLift Servers Anywhere:

- Nel menu principale dell'editor di Unity, scegli Amazon GameLift Serverse seleziona Host with Anywhere. Questa azione apre la pagina del plugin per configurare il gioco con una flotta @Anywhere. La pagina presenta un processo in cinque fasi per integrare, creare e lanciare i componenti del gioco.

Imposta il tuo profilo

Scegli il profilo che desideri utilizzare quando segui questo flusso di lavoro. Il profilo selezionato influisce su tutte le fasi del flusso di lavoro. Tutte le risorse create sono associate all' AWS account del profilo e collocate nella AWS regione predefinita del profilo. Le autorizzazioni dell'utente del profilo determinano l'accesso alle AWS risorse e alle azioni.

1. Seleziona un profilo dall'elenco a discesa dei profili disponibili. Se non hai ancora un profilo o desideri crearne uno nuovo, vai al Amazon GameLift Servers menu e scegli Imposta profili AWS account.
2. Se lo stato di bootstrap non è «Attivo», scegli il profilo Bootstrap e attendi che lo stato passi a «Attivo».

Integra il codice del gioco con l'SDK del server C#

Note

Se hai importato il gioco di esempio, puoi saltare questo passaggio. Le risorse del gioco di esempio dispongono già del codice server e client necessari.

Per questa fase del flusso di lavoro, apporti aggiornamenti al codice client e server del progetto di gioco.

- I server di gioco devono essere in grado di comunicare con il Amazon GameLift Servers servizio per ricevere richieste di avvio di una sessione di gioco, fornire informazioni sulla connessione alla sessione di gioco e segnalare lo stato.
- I client di gioco devono essere in grado di ottenere informazioni sulle sessioni di gioco, partecipare o avviare sessioni di gioco e ottenere informazioni di connessione per partecipare a una partita.

Integra il codice del tuo server

Se stai usando il tuo progetto di gioco con scene personalizzate, usa il codice di esempio fornito per aggiungere il codice server richiesto al tuo progetto di gioco:

1. Nei file di progetto del gioco, apri la Assets/Scripts/Server cartella. Se non esiste, creala.
2. Vai al GitHub repository [aws/ amazon-gamelift-plugin-unity](https://github.com/aws/amazon-gamelift-plugin-unity) e apri il percorso. Samples~/ SampleGame/Assets/Scripts/Server

3. Individua il file `GameLiftServer.cs` e copialo nella cartella del tuo progetto di gioco. Server
Quando crei un eseguibile sul server, usa questo file come obiettivo di compilazione.

Il codice di esempio include questi elementi minimi richiesti, che utilizzano l'SDK del server Amazon GameLift Servers C# (versione 5):

- Inizializza un client API. Amazon GameLift Servers `La InitSDK()` chiamata con i parametri del server è necessaria per una flotta Amazon GameLift Servers Anywhere. Queste impostazioni vengono impostate automaticamente per l'uso nel plug-in.
- Implementa le funzioni di callback richieste per rispondere alle richieste del Amazon GameLift Servers servizio, tra cui `OnStartGameSessionOnProcessTerminate`, e `onHealthCheck`
- Chiamate `ProcessReady()` con una porta designata per avvisare il Amazon GameLift Servers servizio quando il processo del server è pronto per ospitare sessioni di gioco.

Se desideri personalizzare il codice server di esempio, consulta queste risorse:

- [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#)
- [Server C# SDK 5.x per -- Azioni Amazon GameLift Servers](#)

Integra il codice del tuo cliente

Se stai usando il tuo progetto di gioco con scene personalizzate, devi integrare le funzionalità di base nel tuo client di gioco. Devi anche aggiungere elementi dell'interfaccia utente in modo che i giocatori possano accedere e partecipare a una sessione di gioco. Utilizza l'API di servizio per Amazon GameLift Servers (nell' AWS SDK) ottenere informazioni sulla sessione di gioco, creare nuove sessioni di gioco o partecipare a sessioni di gioco esistenti,

Quando crei un client per i test locali con una flotta Anywhere, puoi aggiungere chiamate dirette al Amazon GameLift Servers servizio. Quando sviluppi il tuo gioco per l'hosting su cloud o se prevedi di utilizzare le flotte di Anywhere per l'hosting di produzione, dovrai creare un servizio di backend lato client per gestire tutte le comunicazioni tra i client di gioco e il servizio. Amazon GameLift Servers

Per l'integrazione Amazon GameLift Servers nel codice client, utilizza le seguenti risorse come guida.

- Integra il client con la `GameLiftCoreApi` classe nel GitHub repository `amazon-gamelift-plugin-unity aws/`. Questa classe fornisce controlli per l'autenticazione del giocatore e per il recupero delle informazioni sulla sessione di gioco.

- Visualizza esempi di integrazioni di gioco, disponibili nel repository GitHub `aws/.amazon-gamelift-plugin-unity Samples~/SampleGame/Assets/Scripts/Client/GameLiftClient.cs`
- Segui le istruzioni in **Aggiungi al tuo client Amazon GameLift Servers di gioco Unity**.

Per i client di gioco che si connettono a una flotta Anywhere, il client di gioco necessita delle seguenti informazioni. Il plug-in aggiorna automaticamente il progetto di gioco per utilizzare le risorse create nel plug-in.

- **FleetId** - L'identificatore univoco per la tua flotta Anywhere.
- **FleetLocation** - La posizione personalizzata della tua flotta Anywhere.
- **AwsRegion** - La AWS regione in cui è ospitata la tua flotta Anywhere. Questa è la regione che hai impostato nel tuo profilo utente.
- **ProfileName** - Un profilo di AWS credenziali sul computer locale che consente l'accesso all' AWS SDK per. Amazon GameLift Servers Il client di gioco utilizza queste credenziali per autenticare le richieste al servizio. Amazon GameLift Servers

Note

Il profilo delle credenziali viene generato dal plugin e memorizzato sul computer locale. Di conseguenza, è necessario eseguire il client sul computer locale (o su un computer con lo stesso profilo).

Connect a una flotta Anywhere

In questa fase, designerai una flotta Anywhere da utilizzare. Una flotta Anywhere definisce una raccolta di risorse di elaborazione, che possono essere posizionate ovunque, per l'hosting di server di gioco.

- Se l' AWS account che stai utilizzando attualmente dispone di flotte Anywhere esistenti, apri il campo a discesa **Fleet name** e scegli una flotta. Questo menu a discesa mostra solo le flotte Anywhere nella AWS regione per il profilo utente attualmente attivo.
- Se non ci sono flotte esistenti o desideri crearne una nuova, scegli **Crea una nuova flotta Anywhere** e fornisci un nome per la flotta.

Dopo aver scelto una flotta Anywhere per il tuo progetto, Amazon GameLift Servers verifica che lo stato della flotta sia attivo e visualizza l'ID della flotta. Puoi tenere traccia dello stato di avanzamento di questa richiesta nel registro di output dell'editor Unity.

Registra un calcolo

In questo passaggio, registri la tua workstation locale come risorsa di elaborazione nella nuova flotta Anywhere.

1. Immettete un nome di calcolo per il computer locale. Se aggiungi più di un computer nel parco dati, i nomi devono essere univoci.
2. Scegli Register compute. Puoi tenere traccia dello stato di avanzamento di questa richiesta nel registro di output dell'editor Unity.

Il plugin registra la workstation locale con l'indirizzo IP impostato su localhost (127.0.0.1). Questa impostazione presuppone che il client e il server di gioco vengano eseguiti sulla stessa macchina.

In risposta a questa azione, Amazon GameLift Servers verifica che sia in grado di connettersi al computer e restituisce informazioni sull'elaborazione appena registrata.

Avvia il gioco

In questo passaggio crei i componenti del gioco e li avvii per giocare. Completa le seguenti operazioni:

1. Configura il tuo client di gioco. In questo passaggio, richiedi al plugin di aggiornare una `GameLiftClientSettings` risorsa per il tuo progetto di gioco. Il plugin utilizza questa risorsa per memorizzare determinate informazioni di cui il client di gioco ha bisogno per connettersi al Amazon GameLift Servers servizio.
 - a. Se non hai importato e inizializzato il gioco di esempio, crea una nuova `GameLiftClientSettings` risorsa. Nel menu principale dell'editor Unity, scegli Assets, Create, Amazon GameLift, Client Settings. Se crei più copie del `GameLiftClientSettings` tuo progetto, il plug-in lo rileva automaticamente e ti notifica quale risorsa verrà aggiornata dal plug-in.
 - b. In Launch Game, scegli Configure Client: Apply Anywhere Settings. Questa azione aggiorna le impostazioni del client di gioco per utilizzare la flotta Anywhere che hai appena configurato.
2. Crea ed esegui il tuo client di gioco.

- a. Crea un eseguibile client utilizzando il processo di compilazione standard di Unity. In File, Build Settings, passa la piattaforma a Windows, Mac, Linux. Se hai importato il gioco di esempio e hai inizializzato le impostazioni, l'elenco delle build e l'obiettivo di build vengono aggiornati automaticamente.
 - b. Avvia una o più istanze dell'eseguibile del client di gioco appena creato.
3. Avvia un server di gioco nella tua flotta Anywhere. Scegli Server: Avvia il server nell'editor. Questa attività avvia un server live a cui il client può connettersi finché l'editor Unity rimane aperto.
4. Avvia o partecipa a una sessione di gioco. Nelle istanze del client di gioco, usa l'interfaccia utente per unire ogni client a una sessione di gioco. Il modo in cui esegui questa operazione dipende da come hai aggiunto funzionalità al client.

Se utilizzi il client di gioco di esempio, ha le seguenti caratteristiche:

- Un componente per il login del giocatore. Quando ci si connette a un server di gioco su una flotta Anywhere, non è prevista la convalida del giocatore. Puoi inserire qualsiasi valore per partecipare alla sessione di gioco.
- Una semplice interfaccia utente di accesso al gioco. Quando un client tenta di partecipare a una partita, cerca automaticamente una sessione di gioco attiva con uno slot disponibile. Se non è disponibile alcuna sessione di gioco, il client richiede una nuova sessione di gioco. Se è disponibile una sessione di gioco, il client richiede di partecipare alla sessione di gioco disponibile. Quando provi il gioco con più client simultanei, il primo client avvia la sessione di gioco e i client rimanenti si uniscono automaticamente alla sessione di gioco esistente.
- Sessioni di gioco con quattro slot per giocatori. Puoi avviare fino a quattro istanze del client di gioco contemporaneamente e queste si uniranno alla stessa sessione di gioco.

Avvio da un server eseguibile (opzionale)

Puoi creare e avviare il tuo server di gioco eseguibile per testarlo su una flotta Anywhere.

1. Crea un eseguibile sul server utilizzando il processo di compilazione standard di Unity. In File, Build Settings, passa la piattaforma a Server dedicato e crea.
2. Ottieni un token di autenticazione a breve termine chiamando il comando AWS CLI [get-compute-auth-token](#) con l'ID e AWS la regione della flotta Anywhere. L'ID della flotta viene visualizzato

in Connect to an Anywhere Fleet quando crei la flotta. La AWS regione viene visualizzata in Imposta il tuo profilo quando selezioni il tuo profilo attivo.

```
aws gamelift get-compute-auth-token --fleet-id [your anywhere fleet ID] --region  
[your AWS region]
```

3. Avvia l'eseguibile del server di gioco appena creato da una riga di comando e inserisci un token di autenticazione valido.

```
my_project.exe --authToken [token]
```

Plugin per Unity: distribuisce il tuo gioco su una flotta EC2 gestita

In questo flusso di lavoro, utilizzi il plug-in per preparare il gioco per l'hosting su risorse di elaborazione basate su cloud gestite da Amazon GameLift Servers. Aggiungi il codice di gioco del client e del server per Amazon GameLift Servers aumentarne la funzionalità, quindi carica la build del server sul Amazon GameLift Servers servizio per l'hosting. Una volta completato questo flusso di lavoro, avrai server di gioco in esecuzione nel cloud e un client di gioco funzionante in grado di connettersi ad essi.

Per avviare il flusso di Amazon GameLift Servers lavoro gestito di Amazon EC2:

- Nel menu principale dell'editor Unity, scegli Amazon GameLift Serverse seleziona Host with Managed EC2. Questo flusso di lavoro presenta un processo in sei fasi per integrare, creare, distribuire e avviare i componenti del gioco.

Imposta il tuo profilo

Scegli il profilo che desideri utilizzare quando segui questo flusso di lavoro. Il profilo selezionato influisce su tutte le fasi del flusso di lavoro. Tutte le risorse create sono associate all' AWS account del profilo e collocate nella AWS regione predefinita del profilo. Le autorizzazioni dell'utente del profilo determinano l'accesso alle AWS risorse e alle azioni.

1. Seleziona un profilo dall'elenco a discesa dei profili disponibili. Se non hai ancora un profilo o desideri crearne uno nuovo, vai al Amazon GameLift Servers menu e scegli Imposta profili AWS account.
2. Se lo stato di bootstrap non è «Attivo», scegli il profilo Bootstrap e attendi che lo stato diventi «Attivo».

Integra il tuo gioco con Amazon GameLift Servers

Per questo compito, aggiorni il codice del client e del server nel tuo progetto di gioco.

- I server di gioco devono essere in grado di comunicare con il Amazon GameLift Servers servizio per ricevere richieste di avvio di una sessione di gioco, fornire informazioni sulla connessione alla sessione di gioco e segnalare lo stato.
- I client di gioco devono essere in grado di ottenere informazioni sulle sessioni di gioco, partecipare o avviare sessioni di gioco e ottenere informazioni di connessione per partecipare a una partita.

Note

Se hai importato il gioco di esempio, puoi saltare questo passaggio. Le risorse del gioco di esempio dispongono già del codice server e client necessari.

Integra il codice del tuo server

Quando utilizzi il tuo progetto di gioco con scene personalizzate, usa il codice di esempio fornito per aggiungere il codice server richiesto al tuo progetto di gioco. Se hai integrato il tuo progetto di gioco per testarlo con una flotta Anywhere, hai già completato le istruzioni in questo passaggio.

1. Nei file di progetto del gioco, apri la `Assets/Scripts/Server` cartella. Se non esiste, creala.
2. Vai al GitHub repository [aws/ amazon-gamelift-plugin-unity](https://github.com/aws/amazon-gamelift-plugin-unity) e apri il percorso. `Samples~/SampleGame/Assets/Scripts/Server`
3. Individua il file `GameLiftServer.cs` e copialo nella cartella del tuo progetto di gioco. Server
Quando crei un eseguibile sul server, usa questo file come obiettivo di compilazione.

Il codice di esempio include questi elementi minimi richiesti, che utilizzano l'SDK del server Amazon GameLift Servers C# (versione 5):

- Inizializza un client API. Amazon GameLift Servers La chiamata `initSDK ()` con i parametri del server è necessaria per Amazon GameLift Servers una flotta Anywhere. Queste impostazioni vengono impostate automaticamente per l'uso nel plug-in.
- Implementa le funzioni di callback richieste per rispondere alle richieste del Amazon GameLift Servers servizio, tra cui `OnStartGameSessionOnProcessTerminate`, e `onHealthCheck`

- Chiamate `ProcessReady()` con una porta designata per avvisare il Amazon GameLift Servers servizio quando il processo del server è pronto per ospitare sessioni di gioco.

Se desideri personalizzare il codice server di esempio, consulta queste risorse:

- [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#)
- [Server C# SDK 5.x per -- Azioni Amazon GameLift Servers](#)

Integra il codice del tuo cliente

Per i client di gioco che si connettono a server di gioco basati su cloud, è consigliabile utilizzare un servizio di backend lato client per effettuare chiamate al Amazon GameLift Servers servizio, anziché effettuare le chiamate direttamente dal client di gioco.

Nel flusso di lavoro dei plug-in per l'hosting su una flotta EC2 gestita, ogni scenario di implementazione include un servizio di backend predefinito che include i seguenti componenti:

- Un set di funzioni Lambda e tabelle DynamoDB utilizzate per richiedere sessioni di gioco e recuperare informazioni sulla sessione di gioco. Questi componenti utilizzano un gateway API come proxy.
- Un pool di utenti Amazon Cognito che genera giocatori unici IDs e autentica le connessioni dei giocatori.

Per utilizzare questi componenti, il client di gioco necessita della funzionalità necessaria per inviare richieste al servizio di backend per effettuare le seguenti operazioni:

- Crea un utente giocatore nel pool di utenti di AWS Cognito ed esegui l'autenticazione.
- Partecipa a una sessione di gioco e ricevi informazioni sulla connessione.
- Partecipa a una partita usando matchmaking.

Utilizza le seguenti risorse come guida.

- Integra il client con la [GameLiftCoreApi](#) classe nel GitHub repository [amazon-gamelift-plugin-unityaws/](#). Questa classe fornisce controlli per l'autenticazione del giocatore e per il recupero delle informazioni sulla sessione di gioco.

- [Per visualizzare le integrazioni di gioco di esempio, vai al repository aws/ GitHub „
\[amazon-gamelift-plugin-unity\]\(#\) Samples~/SampleGame/Assets/Scripts/Client/
GameLiftClient.cs](#)
- [Integra la funzionalità del client di Amazon GameLift Servers gioco.](#)

(Solo Unity 6.3+) Integra lo script install.sh

Unity 6.3 su Linux richiede glibc 2.35. Amazon Linux 2023 include glibc 2.34. Viene fornito uno script install.sh che gestisce la compilazione di glibc 2.35 dal sorgente e la configurazione dei binari Unity per utilizzare glibc 2.35.

1. [Vai al repository aws/ e apri il percorso. GitHub amazon-gamelift-plugin-unity
ExtraServerResources](#)
2. Individua il file install.sh e copialo nella cartella contenente la build del server.
3. Se ne hai uno esistente install.sh, unisci il contenuto di entrambi gli script in uno solo.
install.sh

Seleziona lo scenario di implementazione

In questo passaggio, scegli la soluzione di hosting di giochi che desideri implementare in questo momento. Puoi avere più distribuzioni del gioco, utilizzando uno qualsiasi degli scenari.

- Flotta a regione singola: distribuisce il server di gioco su un'unica flotta di risorse di hosting nella regione predefinita del profilo attivo. AWS Questo scenario è un buon punto di partenza per testare l'integrazione del server AWS e la configurazione della build del server. Implementa le seguenti risorse:
 - AWS fleet (On-Demand) con la build del server di gioco installata e funzionante.
 - Pool di utenti e client Amazon Cognito per consentire ai giocatori di autenticarsi e iniziare una partita.
 - Autorizzatore di gateway API che collega il pool di utenti con. APIs
 - Web ACI per limitare le chiamate eccessive dei giocatori al gateway API.
 - API gateway + funzione Lambda per consentire ai giocatori di richiedere uno slot di gioco. Questa funzione chiama `CreateGameSession()` se non ce ne sono disponibili.
 - API gateway + funzione Lambda per consentire ai giocatori di ottenere informazioni di connessione per la loro richiesta di gioco.

- FlexMatch flotta: distribuisce il tuo server di gioco su una serie di flotte e configura un FlexMatch matchmaker con le regole per creare partite tra giocatori. Questo scenario utilizza un hosting Spot a basso costo con una struttura multi-flotta e più sedi per una disponibilità duratura. Questo approccio è utile quando sei pronto per iniziare a progettare un componente matchmaker per la tua soluzione di hosting. In questo scenario, creerai le risorse di base per questa soluzione, che potrai personalizzare in seguito, se necessario. Implementa le seguenti risorse:
 - FlexMatch configurazione del matchmaking e regole di matchmaking impostate per accettare le richieste dei giocatori e formare partite.
 - Tre AWS flotte con la configurazione del server di gioco installata e funzionante in più località. Include due flotte Spot e una flotta On-Demand come backup.
 - AWS coda per il posizionamento delle sessioni di gioco che soddisfa le richieste di partite proposte trovando la migliore risorsa di hosting possibile (in base a fattibilità, costo, latenza dei giocatori, ecc.) e avviando una sessione di gioco.
 - Pool di utenti e client Amazon Cognito per consentire ai giocatori di autenticarsi e iniziare una partita.
 - Autorizzatore di gateway API che collega il pool di utenti con. APIs
 - Web ACI per limitare le chiamate eccessive dei giocatori al gateway API.
 - API gateway + funzione Lambda per consentire ai giocatori di richiedere uno slot di gioco. Questa funzione chiama `StartMatchmaking()`.
 - API gateway + funzione Lambda per consentire ai giocatori di ottenere informazioni di connessione per la loro richiesta di gioco.
 - Tabelle Amazon DynamoDB per archiviare i ticket di matchmaking per i giocatori e le informazioni sulle sessioni di gioco.
 - Argomento SNS + funzione Lambda per GameSessionQueue gestire gli eventi.

Imposta i parametri di gioco

In questo passaggio, descrivi il gioco su cui caricarlo AWS.

- Nome del gioco: fornisci un nome significativo per il tuo progetto di gioco. Questo nome viene utilizzato all'interno del plugin.
- Nome della flotta: fornisci un nome significativo per la tua flotta EC2 gestita. Amazon GameLift Servers utilizza questo nome (insieme all'ID della flotta) per elencare le risorse nella AWS console.

- Nome build: fornisci un nome significativo per la build del server. AWS utilizza questo nome per fare riferimento alla copia della build del server che viene caricata Amazon GameLift Servers e utilizzata per le distribuzioni.
- Parametri di avvio: inserisci le istruzioni opzionali da eseguire all'avvio del server eseguibile su un'istanza gestita della flotta EC2. La lunghezza massima è di 1024 caratteri.
- Cartella del server di gioco: fornisci il percorso di una cartella locale contenente la build del server.
- File del server di gioco: specifica il nome del file eseguibile del server.

Scenario di distribuzione

In questo passaggio, distribuisce il gioco su una soluzione di hosting cloud in base allo scenario di implementazione scelto. Questo processo può richiedere diversi minuti per AWS la convalida della build del server, il provisioning delle risorse di hosting, l'installazione del server di gioco, l'avvio dei processi del server e la preparazione per ospitare sessioni di gioco.

Per iniziare la distribuzione, scegli Deploy. CloudFormation Puoi monitorare lo stato del tuo hosting di giochi qui. Per informazioni più dettagliate, puoi accedere alla console di AWS gestione AWS e visualizzare le notifiche degli eventi. Assicurati di accedere utilizzando lo stesso account, utente e AWS regione del profilo utente attivo nel plug-in.

Una volta completata la distribuzione, il server di gioco è installato su un'istanza AWS EC2. Almeno un processo server è in esecuzione ed è pronto per iniziare una sessione di gioco.

Avvia il client di gioco

Quando la tua flotta sarà dispiegata con successo, ora avrai dei server di gioco attivi e disponibili per ospitare sessioni di gioco. Ora puoi creare il tuo client, avviarlo, connetterti per partecipare alla sessione di gioco.

1. Configura il tuo client di gioco. In questo passaggio, richiedi al plugin di aggiornare una `GameLiftClientSettings` risorsa per il tuo progetto di gioco. Il plugin utilizza questa risorsa per memorizzare determinate informazioni di cui il client di gioco ha bisogno per connettersi al Amazon GameLift Servers servizio.
 - a. Se non hai importato e inizializzato il gioco di esempio, crea una nuova `GameLiftClientSettings` risorsa. Nel menu principale dell'editor Unity, scegli Assets, Create, Amazon GameLift, Client Settings. Se crei più copie del `GameLiftClientSettings` tuo progetto, il plug-in lo rileva automaticamente e ti notifica quale risorsa verrà aggiornata dal plug-in.

- b. In Launch Game, scegli Configura client: applica impostazioni EC2 gestite. Questa azione aggiorna le impostazioni del client di gioco per utilizzare la flotta EC2 gestita che hai appena implementato.
2. Crea il tuo client di gioco. Crea un eseguibile client utilizzando il processo di compilazione standard di Unity. In File, Build Settings, passa la piattaforma a Windows, Mac, Linux. Se hai importato il gioco di esempio e hai inizializzato le impostazioni, l'elenco delle build e l'obiettivo di build vengono aggiornati automaticamente.
3. Avvia l'eseguibile del client di gioco di nuova build. Per iniziare a giocare, avvia da due a quattro istanze client e usa l'interfaccia utente di ciascuna per partecipare a una sessione di gioco.

Se utilizzi il client di gioco di esempio, ha le seguenti caratteristiche:

- Un componente per il login del giocatore. Quando ci si connette a un server di gioco su una flotta Anywhere, non è prevista la convalida del giocatore. Puoi inserire qualsiasi valore per partecipare alla sessione di gioco.
- Una semplice interfaccia utente di accesso al gioco. Quando un client tenta di partecipare a una partita, cerca automaticamente una sessione di gioco attiva con uno slot disponibile. Se non è disponibile alcuna sessione di gioco, il client richiede una nuova sessione di gioco. Se è disponibile una sessione di gioco, il client richiede di partecipare alla sessione di gioco disponibile. Quando provi il gioco con più client simultanei, il primo client avvia la sessione di gioco e i client rimanenti si uniscono automaticamente alla sessione di gioco esistente.
- Sessioni di gioco con quattro slot per giocatori. Puoi avviare fino a quattro istanze del client di gioco contemporaneamente e queste si uniranno alla stessa sessione di gioco.

Plugin per Unity: distribuisce il gioco su una flotta di container gestita

Usa questo flusso di lavoro guidato dai plug-in per creare un'immagine del contenitore per il tuo server di gioco e distribuirla su una soluzione di hosting basata su container. Una volta completato con successo questo flusso di lavoro, il server di gioco containerizzato è in esecuzione nel cloud e puoi utilizzare il plug-in per avviare un client di gioco, connetterti a una sessione di gioco e giocare.

Prima di iniziare

Questo flusso di lavoro presuppone che tu abbia completato le seguenti attività.

- Integra il codice del server di gioco con l'SDK Amazon GameLift Servers del server. Il nostro server di gioco ospitato deve essere in grado di comunicare con il Amazon GameLift Servers servizio in

modo che possa rispondere alle richieste di avvio di nuove sessioni di gioco e segnalare lo stato della sessione di gioco. Se non hai completato questa operazione, ti consigliamo di seguire prima il flusso di lavoro del plug-in Host with Anywhere. Per istruzioni sulla preparazione del codice del server di gioco, consulta [Integra il codice del tuo server](#). Per una flotta di container gestita, devi integrare il gioco con la versione 5.2 o successiva dell'SDK del server.

Note

Se hai importato il gioco di esempio, questa operazione è già stata eseguita per te.

- Package dell'eseguibile del server di gioco per eseguirlo su Linux.
- Raccogli i file da distribuire con la build del tuo server di gioco. Sul computer locale, crea una directory di lavoro per organizzare i file, che verrà incorporata nell'immagine del contenitore del server di gioco. Questi potrebbero includere dipendenze di gioco, uno script per avviare i server di gioco e altri processi all'avvio di un contenitore, ecc.
- Integra il codice del client di gioco con Amazon GameLift Servers. Un modo per completare questa operazione è aggiungere una risorsa di esempio (inclusa nel plug-in) già integrata. Per indicazioni sulla preparazione del codice client di gioco, consulta [Integra il codice del tuo cliente](#).
- Installa Docker sul tuo computer locale. È necessario che questo strumento sia installato se desideri che il plug-in crei immagini dei contenitori per te e le invii a un repository ECR. In alternativa, puoi eseguire queste attività manualmente e indicare al plug-in di utilizzare un'immagine del contenitore esistente. Per ulteriori informazioni sulla creazione manuale dell'immagine, consulta [Creare un'immagine del contenitore per Amazon GameLift Servers](#).

Per avviare il flusso di lavoro dei contenitori Amazon GameLift Servers gestiti:

- Nella barra degli strumenti principale dell'editor Unity, scegli il Amazon GameLift Servers menu e seleziona Contenitori gestiti. Questa azione apre la pagina del plug-in Host with Managed Containers, che presenta un step-by-step processo per creare un'immagine del contenitore con la build del server di gioco, distribuirla in una flotta di container e avviare il gioco.

Passaggio 0: imposta il tuo profilo

Questa sezione mostra il profilo utente attualmente selezionato. Verifica che il profilo utente corrente sia quello che desideri utilizzare per questo flusso di lavoro. Tutte le risorse create in questo flusso

di lavoro sono associate all' AWS account del profilo e collocate nella AWS regione predefinita del profilo. Le autorizzazioni dell'utente del profilo determinano l'accesso alle AWS risorse e alle azioni.

Potrebbe essere necessario modificare il profilo utente selezionato se:

- Al momento non è selezionato alcun profilo.
- Vuoi selezionare un profilo diverso o creare un nuovo profilo.
- È necessario eseguire il bootstrap del profilo selezionato (se lo stato di bootstrap è inattivo).

Per impostare o modificare il profilo utente selezionato

- Nel Amazon GameLift Servers menu, scegli Open AWS Access Credentials.

Fase 1: Valutare la disponibilità del contenitore

Prima di distribuire il tuo server di gioco in una flotta di container, devi impacchettarlo in un'immagine del contenitore e archivarlo in un repository Amazon ECR. Il plug-in può completare queste attività per te oppure puoi eseguirle manualmente. In questo passaggio, fornisci informazioni sullo stato dell'immagine del contenitore e dell'archivio ECR.

Usa le domande di valutazione per indicare al plugin quali passi deve compiere:

- Crea una nuova immagine del contenitore. Se scegli questa opzione, nel passaggio successivo ti verrà richiesta la posizione della directory di build del server di gioco e l'eseguibile della build. Il plugin utilizza un modello Dockerfile (fornito da Amazon GameLift Servers) e lo configura automaticamente per il gioco. È possibile visualizzare il modello all'indirizzo. [Crea un'immagine del contenitore per Amazon GameLift Servers](#) Dopo aver scelto questa opzione, indica dove desideri che il plugin memorizzi la nuova immagine:
- Crea un nuovo repository Amazon ECR e inviaci l'immagine del contenitore. Il plug-in crea un repository ECR privato utilizzando l' AWS account ed è predefinito Regione AWS nel profilo utente selezionato.
- Invia l'immagine del contenitore a un repository Amazon ECR creato in precedenza. Se scegli questa opzione, il passaggio successivo ti chiederà di selezionare un repository Amazon ECR esistente da un elenco. L'elenco include tutti i repository Amazon ECR per l' AWS account e quelli predefiniti Regione AWS nel profilo utente selezionato. Puoi selezionare un repository pubblico o privato.

- Usa un'immagine del contenitore esistente. Se hai creato un'immagine manualmente, ti consigliamo di utilizzare il modello Dockerfile fornito da Amazon GameLift Servers, disponibile all'indirizzo. [Crea un'immagine del contenitore per Amazon GameLift Servers](#) Dopo aver scelto questa opzione, indica dove si trova l'immagine.
- Un'immagine generata da Docker archiviata localmente. Se scegli questa opzione, il plug-in crea un nuovo repository privato Amazon ECR e vi invia il file di immagine locale. Il passaggio successivo richiederà un ID immagine, che il plug-in utilizza per localizzare il file di immagine.
- Un'immagine del contenitore già archiviata in un repository Amazon ECR. Se scegli questa opzione, il passaggio successivo ti chiederà di selezionare un repository e un'immagine Amazon ECR esistenti da un elenco. L'elenco include tutti i repository Amazon ECR per l'AWS account e quelli predefiniti Regione AWS nel profilo utente selezionato. Puoi selezionare un repository pubblico o privato.
- Unity 6.3 su Linux richiede glibc 2.35. Amazon Linux 2023 include glibc 2.34. Di seguito viene fornito un modello Dockerfile che gestisce la compilazione di glibc 2.35 dai sorgenti e la configurazione dei binari Unity per utilizzare glibc 2.35.

Modello Dockerfile per un'immagine del contenitore del server di gioco Unity 6.3

Questo modello contiene le istruzioni minime di cui un contenitore di server di gioco ha bisogno per essere utilizzabile in una flotta. Amazon GameLift Servers Modifica il contenuto in base alle esigenze del tuo server di gioco.

```
# Base image
# -----
# Add the base image that you want to use over here,
# Make sure to use an image with the same architecture as the
# Instance type you are planning to use on your fleets.
FROM public.ecr.aws/amazonlinux/amazonlinux

# Game build directory
# -----
# Add your game build directory in the 'GAME_BUILD_DIRECTORY' env variable below.
#
# Game executable
# -----
# Add the relative path to your executable in the 'GAME_EXECUTABLE' env variable
below.
# The game build provided over here needs to be integrated with gamelift server sdk.
```

```

# This template assumes that the executable path is relative to the game build
directory.
#
# Launch parameters
# -----
# Add any launch parameters to pass into your executable in the 'LAUNCH_PARAMS' env
variable below.
#
# Default directory
# -----
# The value provided in 'HOME_DIR' below will be where the game executable and logs
exist.
#
ARG GAME_BUILD_DIRECTORY
ARG GAME_EXECUTABLE
ARG LAUNCH_PARAMS

ENV GAME_BUILD_DIRECTORY=$GAME_BUILD_DIRECTORY \
    GAME_EXECUTABLE=$GAME_EXECUTABLE \
    LAUNCH_PARAMS=$LAUNCH_PARAMS \
    HOME_DIR="/local/game"

# install dependencies as necessary
RUN yum install -y shadow-utils bison wget gcc make patchelf tar gzip && \
    yum clean all && rm -fr /var/cache

RUN mkdir -p $HOME_DIR
COPY .$GAME_BUILD_DIRECTORY/ $HOME_DIR

# Change directory to home
WORKDIR $HOME_DIR

# Build glibc 2.35 and patch Unity binaries
RUN GLIBC_VERSION="2.35" && \
    INSTALL_PREFIX="/opt/glibc-${GLIBC_VERSION}" && \
    ARCH=$(uname -m) && \
    if [ "$ARCH" = "x86_64" ]; then \
        INTERPRETER="${INSTALL_PREFIX}/lib/ld-linux-x86-64.so.2"; \
        MONO_ARCH="x86_64"; \
    elif [ "$ARCH" = "aarch64" ]; then \
        INTERPRETER="${INSTALL_PREFIX}/lib/ld-linux-aarch64.so.1"; \
        MONO_ARCH="aarch64"; \
    else \
        echo "ERROR: Unsupported architecture: $ARCH"; exit 1; \

```



```

fi && \
cd /tmp && \
wget -q "https://ftp.gnu.org/gnu/glibc/glibc-`${GLIBC_VERSION}`.tar.gz" && \
tar -xzf "glibc-`${GLIBC_VERSION}`.tar.gz" && \
cd "glibc-`${GLIBC_VERSION}`" && mkdir glibc-build && cd glibc-build && \
touch /etc/ld.so.conf && \
../configure --prefix="`${INSTALL_PREFIX}`" && \
make -s all && make -s install && \
cd / && rm -rf /tmp/glibc-* && \
GLIBC_LIB="`${INSTALL_PREFIX}`/lib" && \
EXECUTABLE_NAME=$(echo "$GAME_EXECUTABLE" | sed 's/\.[^\.]*$//') && \
DATA_DIR="$HOME_DIR/`${EXECUTABLE_NAME}`_Data" && \
patchelf --set-interpreter "$INTERPRETER" "$HOME_DIR/`${GAME_EXECUTABLE}`" && \
patchelf --set-rpath "$GLIBC_LIB:$HOME_DIR:/lib64:`${DATA_DIR}`/MonoBleedingEdge/
$MONO_ARCH" "$HOME_DIR/`${GAME_EXECUTABLE}`" && \
patchelf --set-rpath "$GLIBC_LIB:/lib64" "$HOME_DIR/UnityPlayer.so" && \
MONO_LIB="`${DATA_DIR}`/MonoBleedingEdge/`${MONO_ARCH}`/libmonobdwgc-2.0.so" && \
if [ -f "`${MONO_LIB}`" ]; then patchelf --set-rpath "$GLIBC_LIB:/lib64" "`${MONO_LIB}`";
fi && \
GAME_ASSEMBLY="`${HOME_DIR}`/GameAssembly.so" && \
if [ -f "`${GAME_ASSEMBLY}`" ]; then patchelf --set-rpath "$GLIBC_LIB:/lib64"
"`${GAME_ASSEMBLY}`"; fi

RUN useradd -m gamescale && \
echo "gamescale ALL=(ALL) NOPASSWD:ALL" >> /etc/sudoers && \
chown -R gamescale:gamescale $HOME_DIR

# Add permissions to game build
RUN chmod +x ./`${GAME_EXECUTABLE}`

USER gamescale

# Check directory before starting the container
RUN ls -lhrt .

# Check path before starting the container
RUN echo $PATH

# Start the game build
ENTRYPOINT ["/bin/sh", "-c", "`.`/`${GAME_EXECUTABLE}` $LAUNCH_PARAMS"]

```

Fase 2: Configurare la distribuzione delle immagini

In questo passaggio, fornisci le informazioni necessarie al plug-in per distribuire l'immagine del container in una flotta di container. Questo passaggio richiede le seguenti informazioni:

- La posizione della build del server di gioco, dell'immagine del contenitore o del repository Amazon ECR, in base alle selezioni effettuate nella Fase 1.
- Lo scenario da utilizzare per la distribuzione dei container gestiti.
- Impostazioni di distribuzione opzionali. Questa sezione contiene le impostazioni di configurazione che il plugin utilizza per impostazione predefinita. È possibile modificarli o mantenere i valori predefiniti
 - Per impostazione predefinita, il nome del gioco è impostato sul nome del progetto di gioco. Tutte le AWS risorse create dal plugin fanno riferimento al valore del nome del gioco.
 - L'intervallo di porte, il limite di memoria e il limite di vCPU sono impostazioni di configurazione per la flotta di container. Per ulteriori informazioni sulla personalizzazione di questi valori, consulta [Configurare le connessioni di rete](#) l'intervallo delle porte di connessione e [Imposta i limiti delle risorse](#) i limiti delle risorse.
 - Il tag dell'immagine del contenitore viene utilizzato per classificare le immagini del contenitore in Amazon ECR. Il valore predefinito è `unity-gamelift-plugin`.

Opzioni dello scenario di implementazione

Flotta di container in un'unica regione

Questo scenario distribuisce il tuo server di gioco su una singola flotta di container. È un buon punto di partenza per testare l'integrazione del server AWS e la configurazione del container. Implementa le seguenti risorse.

- Amazon GameLift ServersLa definizione del gruppo di container descrive come distribuire ed eseguire le immagini dei container su una flotta di container.
- Amazon GameLift Serversflotta di container (On-Demand) con il container del server di gioco installato e funzionante, con alias.
- Pool di utenti e client Amazon Cognito per consentire ai giocatori di autenticarsi e iniziare una partita.
- Autorizzatore API Gateway che collega il pool di utenti con APIs.

- Elenco di controllo degli accessi Web (ACL) per limitare le chiamate eccessive dei giocatori all'API Gateway.
- Servizio di backend per effettuare richieste al Amazon GameLift Servers servizio per conto dei client di gioco, ad esempio per richiedere sessioni di gioco e partecipare ai giochi:
 - API Gateway + funzione Lambda per consentire ai giocatori di richiedere uno slot per la sessione di gioco. Questa funzione chiama `CreateGameSession()` se non sono disponibili slot aperti.
 - API Gateway + funzione Lambda per consentire ai giocatori di ottenere informazioni di connessione per la loro richiesta di gioco.

Flotta di container a regione singola con FlexMatch

Questo scenario distribuisce il server di gioco su una flotta di container, configura il posizionamento delle sessioni di gioco, e imposta il matchmaking. FlexMatch Questo scenario è utile quando sei pronto per iniziare a progettare un matchmaker personalizzato per la tua soluzione di hosting. Utilizza questo scenario per creare le risorse di base per questa soluzione, che potrai personalizzare in seguito, se necessario. Implementa le seguenti risorse:

- Amazon GameLift Serversdefinizione del gruppo di container che descrive come distribuire ed eseguire le immagini dei container su una flotta di container.
- Amazon GameLift Serversflotta di container (On-Demand) con il container del server di gioco installato e funzionante, con alias.
- FlexMatchconfigurazione del matchmaking e regole di matchmaking impostate per accettare le richieste dei giocatori e formare partite.
- Amazon GameLift Serverscoda di sessione di gioco che soddisfa le richieste di partite proposte trovando la migliore risorsa di hosting possibile (in base a fattibilità, costo, latenza dei giocatori, ecc.) e avviando una sessione di gioco.
- Pool di utenti e client Amazon Cognito per consentire ai giocatori di autenticarsi e iniziare una partita.
- Autorizzatore API Gateway che collega il pool di utenti con APIs.
- Elenco di controllo degli accessi Web (ACL) per limitare le chiamate eccessive dei giocatori all'API Gateway.
- Servizio di backend per effettuare richieste al Amazon GameLift Servers servizio per conto dei client di gioco, ad esempio per richiedere sessioni di gioco e partecipare ai giochi:
 - API Gateway + funzione Lambda per consentire ai giocatori di richiedere uno slot per la sessione di gioco. Questa funzione chiama `StartMatchmaking()` se non sono disponibili slot aperti.

- API Gateway + funzione Lambda per consentire ai giocatori di ottenere informazioni di connessione per la loro richiesta di gioco.
- Tabelle DynamoDB per archiviare i ticket di matchmaking per i giocatori e le informazioni sulle sessioni di gioco.
- Argomento Amazon SNS + funzione Lambda per gestire gli eventi. GameSessionQueue

Implementa una flotta di container

Quando la configurazione della flotta è completa, scegli il pulsante Implementa la flotta di container per avviare l'implementazione. Questo processo può richiedere diversi minuti mentre il plug-in crea un'immagine del container e la invia a ECR, fornisce le risorse di hosting per la flotta di container, distribuisce la flotta e altre AWS risorse per lo scenario della soluzione di hosting selezionata.

Quando si avvia una distribuzione, è possibile monitorare lo stato di avanzamento di ogni fase. A seconda della configurazione, i passaggi potrebbero includere quanto segue:

- Configurazione dell'immagine del contenitore
- Creazione di un repository Amazon ECR
- Creare un'immagine e passare ad Amazon ECR
- Creazione della definizione di un gruppo di contenitori
- Creazione di una flotta di container

Per informazioni più dettagliate sulla distribuzione, scegli Visualizza nella console AWS di gestione. Quando la flotta di container raggiunge lo stato attivo, utilizza attivamente container con processi server pronti per ospitare sessioni di gioco.

Una volta completata l'implementazione, disponi di una flotta di container funzionante pronta per ospitare sessioni di gioco e accettare connessioni tra giocatori.

Non puoi interrompere una distribuzione in corso. Se la distribuzione entra in uno stato errato o fallisce, puoi ricominciare da capo utilizzando l'opzione Reset deployment.

Avvia il client

A questo punto, hai completato tutte le attività per avviare e giocare al gioco multiplayer ospitato con Amazon GameLift Servers. Per giocare, scegli Start Client per avviare un'istanza locale del tuo client di gioco.

- Se hai utilizzato lo scenario a flotta singola, apri un'istanza del client di gioco con un solo giocatore e accedi alla mappa del server per muoverti. Puoi aprire una seconda istanza del client di gioco per aggiungere un secondo giocatore alla stessa mappa di gioco del server.
- Se hai implementato FlexMatch lo scenario, la soluzione di hosting attende che almeno due client di gioco effettuino richieste di matchmaking. Apri almeno due istanze del tuo client di gioco con un giocatore. I due giocatori verranno abbinati e verrà richiesto di partecipare a una sessione di gioco per la partita.

Aggiorna una flotta di container

Se hai implementato con successo una soluzione di hosting di container gestiti, puoi utilizzare la funzionalità Update deployment. Questa opzione consente di aggiornare le impostazioni di configurazione per una flotta di container distribuita, senza dover creare una nuova flotta.

Quando aggiorni una distribuzione, puoi distribuire un'immagine del contenitore con una build diversa del server di gioco, modificare il repository Amazon ECR, scegliere uno scenario di distribuzione diverso e personalizzare le impostazioni di configurazione opzionali.

Quando sei pronto per distribuire le modifiche, scegli **Aggiorna**. Il tempo necessario per un aggiornamento della distribuzione è simile a quello di una distribuzione completa. Per informazioni dettagliate sulla distribuzione, scegli **Visualizza** nella console AWS di gestione.

Pulisci le risorse distribuite

Come best practice, pulisci AWS le risorse per la tua soluzione di container gestiti non appena non ne hai più bisogno. Potresti continuare a sostenere dei costi per queste risorse se non le rimuovi.

Elimina le seguenti risorse:

- Stack di risorse del contenitore gestito. Le risorse in questo stack dipendono dallo scenario di distribuzione selezionato. Per eliminare l'intero stack, usa la CloudFormation console. Gli stack generati dal Amazon GameLift Servers plugin utilizzano la seguente convenzione di denominazione: `GameLiftPluginForUnity-{GameName}-Containers`. Attendi il completamento del processo di eliminazione dello stack prima di avviare una nuova distribuzione di contenitori gestiti nel plug-in. Per ulteriori informazioni, consulta [Eliminare uno stack dalla console](#). CloudFormation
- Repository Amazon ECR. Se hai utilizzato il plug-in per creare un repository per l'immagine del contenitore, potresti voler eliminare tutti i repository che non sono più necessari. Non è necessario

eliminare un repository prima di reimpostare la distribuzione di contenitori gestiti. Se aggiorni o ripristini una distribuzione, il plug-in utilizzerà automaticamente lo stesso repository a meno che non venga richiesto di utilizzarne un altro. Per ulteriori informazioni, consulta [Eliminazione di un repository privato in Amazon ECR](#).

Amazon GameLift Serversplugin per Unity per server SDK 4

Note

Questo argomento fornisce informazioni su una versione precedente del Amazon GameLift Servers plug-in per Unity. La versione 1.x utilizza l'SDK del server per Amazon GameLift Servers 4.x o versioni precedenti. Per la documentazione sulla versione più recente del plug-in, che utilizza il server SDK 5.x e supporta funzionalità più recenti come Amazon GameLift Servers Anywhere e l'hosting gestito di container, consulta [Amazon GameLift Serversplugin per Unity \(server SDK 5.x\)](#)

Amazon GameLift Serversfornisce strumenti per preparare i server di gioco multiplayer su cui funzionare. Amazon GameLift Servers Il Amazon GameLift Servers plug-in per Unity semplifica l'Amazon GameLift Serversintegrazione nei tuoi progetti di gioco Unity e la distribuzione di Amazon GameLift Servers risorse per l'hosting su cloud. Usa il plug-in per Unity per accedere Amazon GameLift Servers APIs e distribuire AWS CloudFormation modelli per scenari di gioco comuni.

Dopo aver configurato il plug-in, puoi provare l'[esempio di Amazon GameLift Servers Unity](#) su GitHub.

Argomenti

- [Integrazione Amazon GameLift Servers con un progetto di server di gioco Unity](#)
- [Integrazione Amazon GameLift Servers con un progetto client di gioco Unity](#)
- [Installa e configura il plugin](#)
- [Prova il gioco localmente](#)
- [Implementa uno scenario](#)
- [Integra i giochi con Unity Amazon GameLift Servers](#)
- [Importa ed esegui un gioco di esempio](#)

Integrazione Amazon GameLift Servers con un progetto di server di gioco Unity

Note

Questo argomento fornisce informazioni su una versione precedente del Amazon GameLift Servers plug-in per Unity. La versione 1.x utilizza l'SDK del server per Amazon GameLift Servers 4.x o versioni precedenti. Per la documentazione sulla versione più recente del plug-in, che utilizza il server SDK 5.x e supporta funzionalità più recenti come Amazon GameLift Servers Anywhere e l'hosting gestito di container, consulta. [Amazon GameLift Servers plugin per Unity \(server SDK 5.x\)](#)

Questo argomento ti aiuta a preparare il tuo server di gioco personalizzato su cui ospitare. Amazon GameLift Servers Il server di gioco deve essere in grado Amazon GameLift Servers di notificare il suo stato, di avviare e interrompere le sessioni di gioco quando richiesto e di eseguire altre attività. Per ulteriori informazioni, consulta [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#).

Prerequisiti

Prima di integrare il server di gioco, completa le seguenti attività:

- [Configurare un ruolo di servizio IAM per Amazon GameLift Servers](#)
- [Installa e configura il plugin](#)

Configura un nuovo processo del server

Note

Questo argomento si riferisce al Amazon GameLift Servers plug-in per Unity versione 1.0.0, che utilizza il server SDK 4.x o precedente.

Imposta la comunicazione con Amazon GameLift Servers e segnala che il processo del server è pronto per ospitare una sessione di gioco.

1. Inizializza l'SDK del server chiamando. `InitSDK()`
2. Per preparare il server ad accettare una sessione di gioco, chiama `ProcessReady()` indicando la porta di connessione e i dettagli sulla posizione della sessione di gioco. Includi

i nomi delle funzioni di callback richiamate dal Amazon GameLift Servers servizio, ad esempio, `OnGameSession()`, `OnGameSessionUpdate()`, `OnProcessTerminate()`, `OnHealthCheck()`. Amazon GameLift Servers potrebbero essere necessari alcuni minuti per fornire una richiamata.

3. Amazon GameLift Servers aggiorna lo stato del processo del server su `ACTIVE`.
4. Amazon GameLift Servers chiama `onHealthCheck` periodicamente.

Il seguente esempio di codice mostra come configurare un semplice processo server con Amazon GameLift Servers.

```
//initSDK
var initSDKOutcome = GameLiftServerAPI.InitSDK();

//processReady
// Set parameters and call ProcessReady
var processParams = new ProcessParameters(
    this.OnGameSession,
    this.OnProcessTerminate,
    this.OnHealthCheck,
    this.OnGameSessionUpdate,
    port,
    // Examples of log and error files written by the game server
    new LogParameters(new List<string>()
        {
            "C:\\game\\logs",
            "C:\\game\\error"
        })
);

var processReadyOutcome = GameLiftServerAPI.ProcessReady(processParams);

// Implement callback functions
void OnGameSession(GameSession gameSession)
{
    // game-specific tasks when starting a new game session, such as loading map
    // When ready to receive players
    var activateGameSessionOutcome = GameLiftServerAPI.ActivateGameSession();
}

void OnProcessTerminate()
{
```



```
// game-specific tasks required to gracefully shut down a game session,  
// such as notifying players, preserving game state data, and other cleanup  
var ProcessEndingOutcome = GameLiftServerAPI.ProcessEnding();  
}  
  
bool OnHealthCheck()  
{  
    bool isHealthy;  
    // complete health evaluation within 60 seconds and set health  
    return isHealthy;  
}
```

Inizia una sessione di gioco

Note

Questo argomento si riferisce al Amazon GameLift Servers plug-in per Unity versione 1.0.0, che utilizza il server SDK 4.x o precedente.

Una volta completata l'inizializzazione del gioco, puoi iniziare una sessione di gioco.

1. Implementa la funzione di callback. `onStartGameSession` Amazon GameLift Servers chiama questo metodo per avviare una nuova sessione di gioco sul processo server e ricevere le connessioni dei giocatori.
2. Per attivare una sessione di gioco, chiama. `ActivateGameSession()` Per ulteriori informazioni sull'SDK, consulta [SDK del server C# per Amazon GameLift Servers 4.x -- Azioni](#).

Il seguente esempio di codice illustra come avviare una sessione di gioco con. Amazon GameLift Servers

```
void OnStartGameSession(GameSession gameSession)  
{  
    // game-specific tasks when starting a new game session, such as loading map  
    ...  
    // When ready to receive players  
    var activateGameSessionOutcome = GameLiftServerAPI.ActivateGameSession();  
}
```

Termina una sessione di gioco

Note

Questo argomento si riferisce al Amazon GameLift Servers plug-in per Unity versione 1.0.0, che utilizza il server SDK 4.x o precedente.

Notifica Amazon GameLift Servers quando una sessione di gioco sta per terminare. Come best practice, chiudi i processi del server al termine delle sessioni di gioco per riciclare e aggiornare le risorse di hosting.

1. Imposta una funzione denominata `onProcessTerminate` per ricevere richieste Amazon GameLift Servers e chiamate. `ProcessEnding()`
2. Lo stato del processo cambia `inTERMINATED`.

L'esempio seguente descrive come terminare un processo per una sessione di gioco.

```
var processEndingOutcome = GameLiftServerAPI.ProcessEnding();

if (processReadyOutcome.Success)
    Environment.Exit(0);

// otherwise, exit with error code
Environment.Exit(errorCode);
```

Crea server, costruisci e carica su Amazon GameLift Servers

Note

Questo argomento si riferisce al Amazon GameLift Servers plug-in per Unity versione 1.0.0, che utilizza il server SDK 4.x o precedente.

Dopo aver integrato il server di gioco con Amazon GameLift Servers, carica i file di build su una flotta in modo che Amazon GameLift Servers possa utilizzarli per l'hosting dei giochi. Per ulteriori informazioni su come caricare il server su Amazon GameLift Servers, consulta [Crea un server di gioco costruito per Amazon GameLift Servers](#).

Integrazione Amazon GameLift Servers con un progetto client di gioco Unity

Note

Questo argomento fornisce informazioni su una versione precedente del Amazon GameLift Servers plug-in per Unity. La versione 1.x utilizza l'SDK del server per Amazon GameLift Servers 4.x o versioni precedenti. Per la documentazione sulla versione più recente del plug-in, che utilizza il server SDK 5.x e supporta funzionalità più recenti come Amazon GameLift Servers Anywhere e l'hosting gestito di container, consulta. [Amazon GameLift Servers plugin per Unity \(server SDK 5.x\)](#)

Questo argomento ti aiuta a configurare un client di gioco per connetterti alle sessioni di gioco Amazon GameLift Servers ospitate tramite un servizio di backend. Utilizzalo Amazon GameLift Servers APIs per avviare il matchmaking, richiedere il posizionamento della sessione di gioco e altro ancora.

Aggiungi codice al progetto del servizio di backend per consentire la comunicazione con il servizio. Amazon GameLift Servers Un servizio di backend gestisce tutte le comunicazioni del client di gioco con il GameLift servizio. Per ulteriori informazioni sui servizi di backend, consulta.

Un server di backend gestisce le seguenti attività del client di gioco:

- Personalizza l'autenticazione per i tuoi giocatori.
- Richiedere informazioni su sessioni di gioco attive dal servizio Amazon GameLift Servers.
- Crea una nuova sessione di gioco.
- Aggiungi un giocatore a una sessione di gioco esistente.
- Rimuovi un giocatore da una sessione di gioco esistente.

Argomenti

- [Prerequisiti](#)
- [Inizializza un client di gioco](#)
- [Crea una sessione di gioco su una flotta specifica](#)
- [Aggiungere giocatori alle sessioni di gioco](#)
- [Rimuovere un giocatore da una sessione di gioco](#)

Prerequisiti

Prima di configurare la comunicazione del server di gioco con il Amazon GameLift Servers client, completa le seguenti attività:

- [Configurare un account AWS utente](#)
- [Installa e configura il plugin](#)
- [Integrazione Amazon GameLift Servers con un progetto di server di gioco Unity](#)
- [Implementa flotte di hosting per Amazon GameLift Servers](#)

Inizializza un client di gioco

Note

Questo argomento si riferisce al Amazon GameLift Servers plug-in per Unity versione 1.0.0, che utilizza il server SDK 4.x o precedente.

Aggiungi codice per inizializzare un client di gioco. Esegui questo codice all'avvio, è necessario per altre Amazon GameLift Servers funzioni.

1. `InizializzaAmazonGameLiftClient`. Chiama `AmazonGameLiftClient` con una configurazione client predefinita o una configurazione personalizzata. Per ulteriori informazioni su come configurare un client, vedere [Configura l'Amazon GameLift Servers API](#).
2. Genera un ID giocatore univoco per ogni giocatore per connettersi a una sessione di gioco. Per ulteriori informazioni, consulta [Genera giocatore IDs](#).

Gli esempi seguenti mostrano come configurare un Amazon GameLift Servers client.

```
public class GameLiftClient
{
    private GameLift gl;
    //A sample way to generate random player IDs.
    bool includeBrackets = false;
    bool includeDashes = true;
    string playerId = AZ::Uuid::CreateRandom().ToString<string>(includeBrackets,
includeDashes);
}
```

```
private Amazon.GameLift.Model.PlayerSession psession = null;
public AmazonGameLiftClient aglc = null;

public void CreateGameLiftClient()
{
    //Access Amazon GameLift Servers service by setting up a configuration.
    //The default configuration specifies a location.
    var config = new AmazonGameLiftConfig();
    config.RegionEndpoint = Amazon.RegionEndpoint.USEast1;

    CredentialProfile profile = null;
    var nscf = new SharedCredentialsFile();
    nscf.TryGetProfile(profileName, out profile);
    AWSCredentials credentials = profile.GetAWSCredentials(null);
    //Initialize Amazon GameLift Servers Client with default client
    configuration.
        aglc = new AmazonGameLiftClient(credentials, config);
}
}
```

Crea una sessione di gioco su una flotta specifica

Note

Questo argomento si riferisce al Amazon GameLift Servers plug-in per la versione 1.0.0 di Unity, che utilizza il server SDK 4.x o precedente.

Aggiungere codice per avviare nuove sessioni di gioco nei parchi di istanze distribuite e renderle disponibili ai giocatori. Dopo Amazon GameLift Servers aver creato la nuova sessione di gioco e aver restituito un `GameSession`, puoi aggiungervi giocatori.

- Effettua una richiesta per una nuova sessione di gioco.
 - Se il gioco utilizza flotte, chiama `CreateGameSession()` con un ID flotta o alias, un nome di sessione e il numero massimo di giocatori simultanei per la partita.
 - Se il gioco utilizza code, chiama `StartGameSessionPlacement()`

L'esempio seguente mostra come creare una sessione di gioco.

```
public Amazon.GameLift.Model.GameSession()
{
    var cgsreq = new Amazon.GameLift.Model.CreateGameSessionRequest();
    //A unique identifier for the alias with the fleet to create a game session in.
    cgsreq.AliasId = aliasId;
    //A unique identifier for a player or entity creating the game session
    cgsreq.CreatorId = playerId;
    //The maximum number of players that can be connected simultaneously to the game
    session.
    cgsreq.MaximumPlayerSessionCount = 4;

    //Prompt an available server process to start a game session and retrieves
    connection information for the new game session
    Amazon.GameLift.Model.CreateGameSessionResponse cgsres =
    aglc.CreateGameSession(cgsreq);
    string gsid = cgsres.GameSession != null ? cgsres.GameSession.GameSessionId : "N/
    A";
    Debug.Log((int)cgsres.HttpStatusCode + " GAME SESSION CREATED: " + gsid);
    return cgsres.GameSession;
}
```

Aggiungere giocatori alle sessioni di gioco

Note

Questo argomento si riferisce al Amazon GameLift Servers plug-in per Unity versione 1.0.0, che utilizza il server SDK 4.x o precedente.

Dopo Amazon GameLift Servers aver creato la nuova sessione di gioco e aver restituito un `GameSession` oggetto, puoi aggiungervi giocatori.

1. Riserva uno slot per giocatori in una sessione di gioco creando una nuova sessione di gioco. Usa `CreatePlayerSession` o `CreatePlayerSessions` con l'ID della sessione di gioco e un ID univoco per ogni giocatore.
2. Connect alla sessione di gioco. Recupera l'`PlayerSession` oggetto per ottenere le informazioni di connessione della sessione di gioco. Puoi usare queste informazioni per stabilire una connessione diretta al processo del server:
 - a. Utilizzate la porta specificata e il nome DNS o l'indirizzo IP del processo server.

- b. Usa il nome DNS e la porta delle tue flotte. Il nome e la porta DNS sono obbligatori se le tue flotte hanno abilitato la generazione di certificati TLS.
- c. Fai riferimento all'ID della sessione del giocatore. L'ID della sessione del giocatore è necessario se il server di gioco convalida le connessioni dei giocatori in entrata.

Gli esempi seguenti mostrano come riservare un posto giocatore in una sessione di gioco.

```
public Amazon.GameLift.Model.PlayerSession
CreatePlayerSession(Amazon.GameLift.Model.GameSession gsession)
{
    var cpsreq = new Amazon.GameLift.Model.CreatePlayerSessionRequest();
    cpsreq.GameSessionId = gsession.GameSessionId;
    //Specify game session ID.
    cpsreq.PlayerId = playerId;
    //Specify player ID.
    Amazon.GameLift.Model.CreatePlayerSessionResponse cpsres =
aglc.CreatePlayerSession(cpsreq);
    string psid = cpsres.PlayerSession != null ? cpsres.PlayerSession.PlayerSessionId :
"N/A";
    return cpsres.PlayerSession;
}
```

Il codice seguente illustra come connettere un giocatore alla sessione di gioco.

```
public bool ConnectPlayer(int playerIdX, string playerSessionId)
{
    //Call ConnectPlayer with player ID and player session ID.
    return server.ConnectPlayer(playerIdx, playerSessionId);
}
```

Rimuovere un giocatore da una sessione di gioco

Note

Questo argomento si riferisce al Amazon GameLift Servers plug-in per la versione 1.0.0 di Unity, che utilizza il server SDK 4.x o precedente.

Puoi rimuovere i giocatori dalla sessione di gioco quando escono dal gioco.

1. Notifica al Amazon GameLift Servers servizio che un giocatore si è disconnesso dal processo del server. Chiama `RemovePlayerSession` con l'ID di sessione del giocatore.
2. Verifica che venga `RemovePlayerSession` restituito `Success`. Quindi, Amazon GameLift Servers cambia lo slot del giocatore rendendolo disponibile, che Amazon GameLift Servers può assegnare a un nuovo giocatore.

L'esempio seguente illustra come rimuovere una sessione di gioco.

```
public void DisconnectPlayer(int playerIdx)
{
    //Receive the player session ID.
    string playerIdSessionId = playerSessions[playerIdx];
    var outcome = GameLiftServerAPI.RemovePlayerSession(playerSessionId);
    if (outcome.Success)
    {
        Debug.Log (":) PLAYER SESSION REMOVED");
    }
    else
    {
        Debug.Log(":(PLAYER SESSION REMOVE FAILED. RemovePlayerSession()
        returned " + outcome.Error.ToString());
    }
}
```

Installa e configura il plugin

Note

Questo argomento si riferisce al Amazon GameLift Servers plug-in per Unity versione 1.0.0, che utilizza il server SDK 4.x o precedente.

Questa sezione descrive come scaricare, installare e configurare il Amazon GameLift Servers plug-in per Unity, versione 1.0.0.

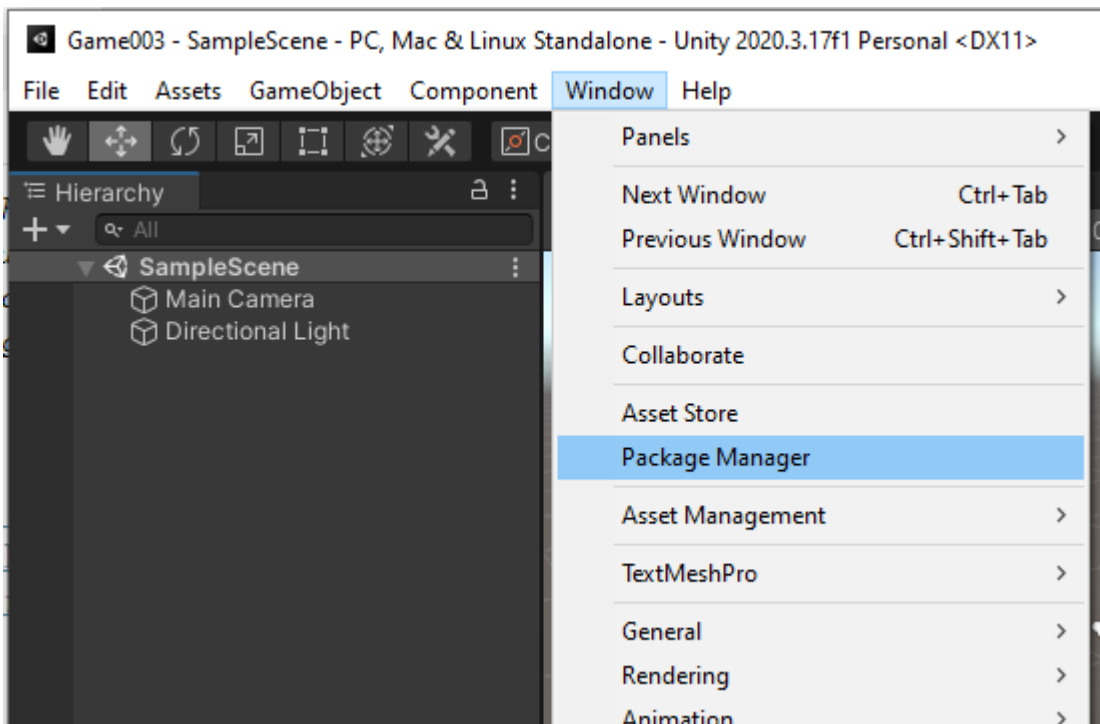
Prerequisiti

- Unity per Windows 2019.4 LTS, Windows 2020.3 LTS o Unity per macOS
- Versione attuale di Java

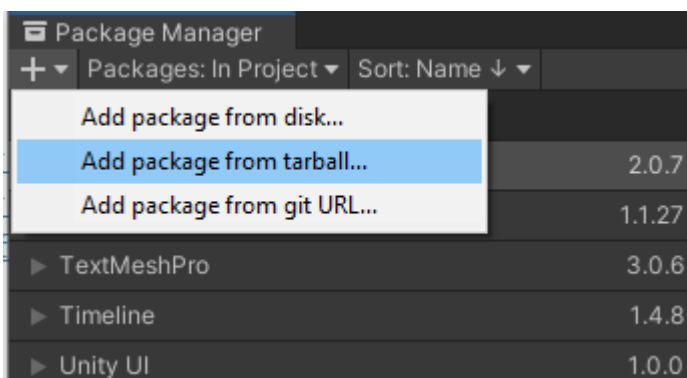
- Versione attuale di .NET 4.x

Per scaricare e installare il plugin per Unity

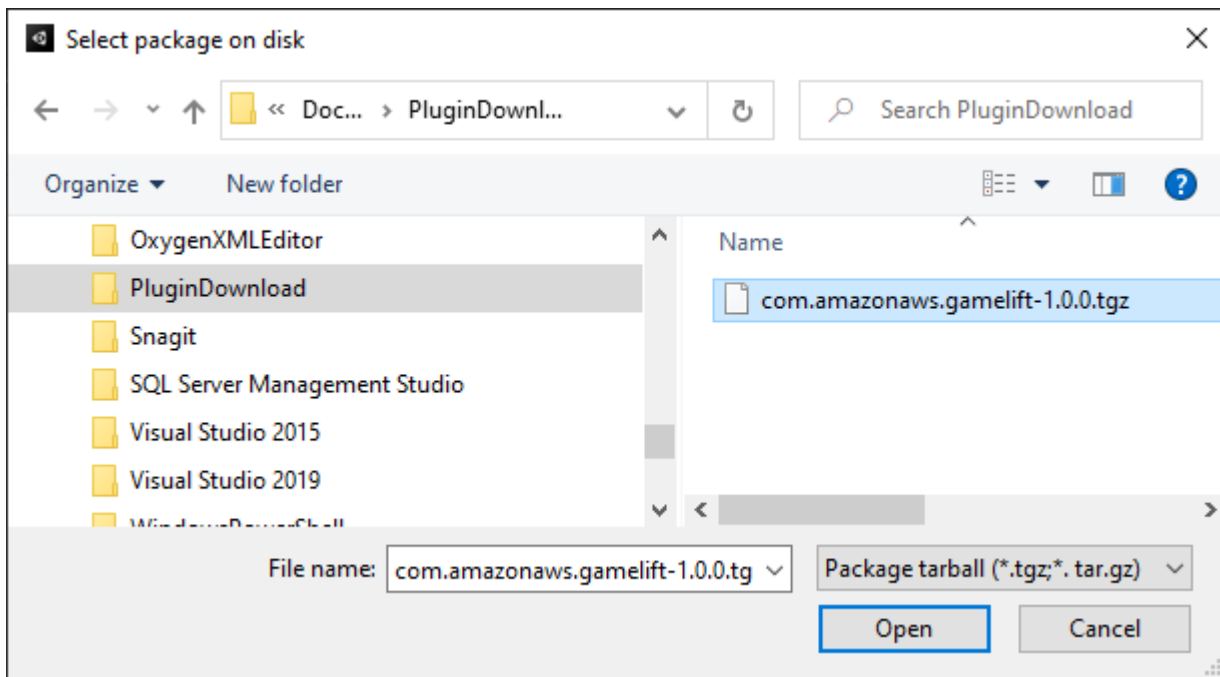
1. Scarica il Amazon GameLift Servers plugin per Unity. Puoi trovare l'ultima versione nella pagina del [repository del Amazon GameLift Servers plugin per Unity](#). Nella [versione più recente](#), scegli Risorse, quindi scarica il `com.amazonaws.gamelift-version.tgz` file.
2. Avvia Unity e scegli un progetto.
3. Nella barra di navigazione in alto, sotto Finestra scegli Package Manager:



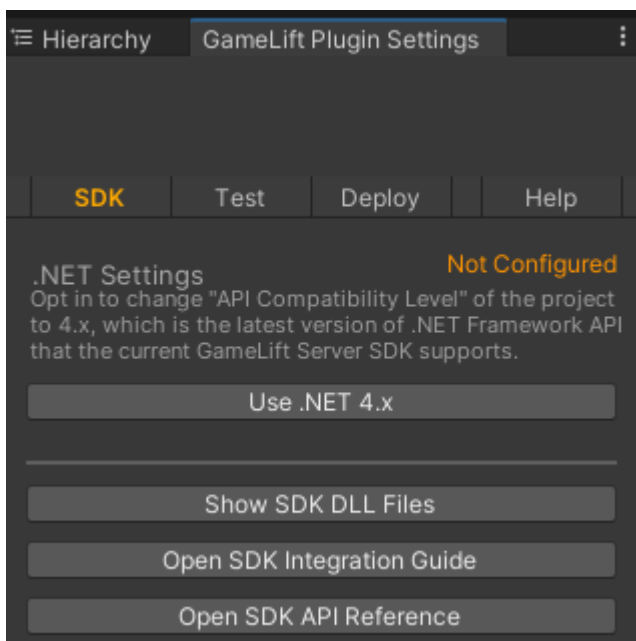
4. Nella scheda Package Manager scegli +, quindi scegli Aggiungi pacchetto da tarball... :



5. Nella finestra Seleziona pacchetti su disco, accedi alla `com.amazonaws.gamelift` cartella, scegli il file `com.amazonaws.gamelift-version.tgz`, quindi scegli Apri:



6. Dopo che Unity ha caricato il plug-in, Amazon GameLift Servers appare come una nuova voce nel menu Unity. Potrebbero essere necessari alcuni minuti per installare e ricompilare gli script. La scheda Impostazioni del Amazon GameLift Servers plugin si apre automaticamente.



7. Nel riquadro SDK, scegli Use.NET 4.x.

Una volta configurato, lo stato cambia da Non configurato a Configurato.

Prova il gioco localmente

Note

Questo argomento si riferisce al Amazon GameLift Servers plugin per Unity versione 1.0.0, che utilizza il server SDK 4.x o precedente.

Usa Amazon GameLift Servers Local per eseguirlo Amazon GameLift Servers sul tuo dispositivo locale. Puoi usare Amazon GameLift Servers Local per verificare le modifiche al codice in pochi secondi, senza una connessione di rete.

Configura i test locali

1. Nella finestra del plugin per Unity, scegli la scheda Test.
2. Nel riquadro Test, scegli Download Amazon GameLift Servers locale. Il plugin per Unity apre una finestra del browser e scarica il `GameLift_06_03_2021.zip` file nella cartella dei download.

Il download include l'SDK del server C#, i file di sorgente.NET e un component.NET compatibile con Unity.

3. Decomprimere il file `GameLift_06_03_2021.zip` scaricato.
4. Nella finestra Impostazioni del Amazon GameLift Servers plug-in, scegli Percorso Amazon GameLift Servers locale, vai alla cartella decompressa, scegli il file **GameLiftLocal.jar**, quindi scegli Apri.

Una volta configurato, lo stato del test locale cambia da Non configurato a Configurato.

5. Verifica lo stato del JRE. Se lo stato è Non configurato, scegli Scarica JRE e installa la versione Java consigliata.

Dopo aver installato e configurato l'ambiente Java, lo stato cambia in Configurato.

Esegui il gioco locale

1. Nella scheda del plugin per Unity, scegli la scheda Test.
2. Nel riquadro Test, scegli Open Local Test UI.
3. Nella finestra Local Testing, specifica un percorso eseguibile del server. Seleziona... per selezionare il percorso e il nome eseguibile dell'applicazione server.

4. Nella finestra Local Testing, specifica una porta GL Local.
5. Scegli Deploy & Run per distribuire ed eseguire il server.
6. Per arrestare il server di gioco, scegli Stop o chiudi le finestre del server di gioco.

Implementa uno scenario

Note

Questo argomento fa riferimento al Amazon GameLift Servers plug-in per Unity versione 1.0.0, che utilizza il server SDK 4.x o precedente.

Uno scenario utilizza un CloudFormation modello per creare le risorse necessarie per implementare una soluzione di hosting cloud per il gioco. Questa sezione descrive gli scenari Amazon GameLift Servers forniti e come utilizzarli.

Prerequisiti

Per implementare lo scenario, è necessario un ruolo IAM per il Amazon GameLift Servers servizio. Per informazioni su come creare un ruolo per Amazon GameLift Servers, consulta [Configurare un account AWS utente](#).

Ogni scenario richiede le autorizzazioni per le seguenti risorse:

- Amazon GameLift Servers
- Simple Storage Service (Amazon S3)
- CloudFormation
- Gateway API
- AWS Lambda
- AWS WAFV2
- Amazon Cognito

Scenari

Note

Questo argomento fa riferimento al Amazon GameLift Servers plug-in per Unity versione 1.0.0, che utilizza il server SDK 4.x o precedente.

Il Amazon GameLift Servers plug-in per Unity include i seguenti scenari:

Solo autenticazione

Questo scenario crea un servizio di backend di gioco che esegue l'autenticazione dei giocatori senza la funzionalità del server di gioco. Il modello crea le seguenti risorse nel tuo account:

- Un pool di utenti Amazon Cognito per archiviare le informazioni di autenticazione dei giocatori.
- Un AWS Lambda gestore basato su endpoint REST di Amazon API Gateway che avvia giochi e visualizza le informazioni sulla connessione del gioco.

Flotta per una singola regione

Questo scenario crea un servizio di backend di gioco con un'unica Amazon GameLift Servers flotta. Crea le seguenti risorse:

- Un pool di utenti Amazon Cognito per consentire a un giocatore di autenticarsi e iniziare una partita.
- Un AWS Lambda gestore per la ricerca di una sessione di gioco esistente con uno slot per giocatori aperto sulla flotta. Se non riesce a trovare uno slot libero, crea una nuova sessione di gioco.

Flotta multiregionale con coda e matchmaker personalizzato

Questo scenario crea partite utilizzando Amazon GameLift Servers code e un matchmaker personalizzato per raggruppare i giocatori più anziani nel pool d'attesa. Crea le seguenti risorse:

- Un argomento di Amazon Simple Notification Service su cui vengono Amazon GameLift Servers pubblicati messaggi. Per ulteriori informazioni sugli argomenti e sulle notifiche SNS, consulta [Imposta la notifica degli eventi per il posizionamento della sessione di gioco](#)

- Una funzione Lambda richiamata dal messaggio che comunica i dettagli del posizionamento e della connessione al gioco.
- Una tabella Amazon DynamoDB per memorizzare i dettagli del posizionamento e della connessione al gioco. `GetGameConnection` le chiamate vengono lette da questa tabella e restituiscono le informazioni di connessione al client di gioco.

Individua le flotte con una coda e un matchmaker personalizzato

Questo scenario crea partite utilizzando Amazon GameLift Servers code e un matchmaker personalizzato e configura tre flotte. Crea le seguenti risorse:

- Due flotte Spot che contengono diversi tipi di istanze per garantire la durabilità in caso di indisponibilità di Spot.
- Una flotta On-Demand che funge da backup per le altre flotte Spot. Per ulteriori informazioni sulla progettazione delle flotte, consulta [Personalizzazioni delle risorse di hosting](#)
- Una Amazon GameLift Servers coda per mantenere alta la disponibilità dei server e ridurre i costi. Per ulteriori informazioni e procedure consigliate sulle code, vedere [Personalizza una coda di sessioni di gioco](#)

FlexMatch

Questo scenario utilizza FlexMatch, un servizio di matchmaking gestito, per abbinare i giocatori. Per ulteriori informazioni su FlexMatch, vedi [What is Amazon GameLift Servers FlexMatch](#). Questo scenario crea le seguenti risorse:

- Una funzione Lambda per creare un ticket di matchmaking dopo aver ricevuto le richieste. `StartGame`
- Una funzione Lambda separata per ascoltare gli eventi delle FlexMatch partite.

Per evitare costi inutili Account AWS, rimuovi le risorse create da ogni scenario dopo averle utilizzate. Elimina lo CloudFormation stack corrispondente.

Aggiorna le credenziali AWS

Note

Questo argomento si riferisce al Amazon GameLift Servers plug-in per Unity versione 1.0.0, che utilizza il server SDK 4.x o precedente.

Il Amazon GameLift Servers plug-in per Unity richiede credenziali di sicurezza per distribuire uno scenario. È possibile creare nuove credenziali o utilizzare credenziali esistenti.

Per ulteriori informazioni sulla configurazione delle credenziali, consulta [Comprendere e](#) ottenere le credenziali. AWS

Per aggiornare le credenziali AWS

1. In Unity, nel plugin per la scheda Unity, scegli la scheda Deploy.
2. Nel riquadro Distribuisci, scegli AWS Credenziali.
3. Puoi creare nuove AWS credenziali o scegliere credenziali esistenti.
 - Per creare credenziali, scegli Crea nuovo profilo di credenziali, quindi specifica il nuovo nome del profilo, l'ID della chiave di AWS accesso, la chiave AWS segreta e. Regione AWS
 - Per scegliere le credenziali esistenti, scegli il profilo Scegli le credenziali esistenti, quindi seleziona un nome di profilo e. Regione AWS
4. Nella finestra Aggiorna AWS credenziali, scegli Aggiorna profilo credenziali.

Aggiorna il bootstrap dell'account

Note

Questo argomento si riferisce al Amazon GameLift Servers plug-in per Unity versione 1.0.0, che utilizza il server SDK 4.x o precedente.

La posizione di bootstrap è un bucket Amazon S3 utilizzato durante la distribuzione. Viene utilizzato per archiviare le risorse dei server di gioco e altre dipendenze. Il bucket Regione AWS scelto deve corrispondere alla stessa regione che utilizzerai per la distribuzione dello scenario.

Per ulteriori informazioni sui bucket Amazon S3, consulta [Creazione, configurazione e utilizzo dei bucket Amazon Simple Storage Service](#).

Per aggiornare la posizione di avvio dell'account

1. In Unity, nel plugin per la scheda Unity, scegli la scheda Deploy.
2. Nel riquadro Deploy, scegli Update Account Bootstrap.
3. Nella finestra Account Bootstrapping, scegli un bucket Amazon S3 esistente o creane uno nuovo:
 - Per scegliere un bucket esistente, scegli Scegli un bucket Amazon S3 esistente e Aggiorna per salvare la selezione.
 - Scegli Crea nuovo bucket Amazon S3 per creare un nuovo bucket Amazon Simple Storage Service, quindi scegli una policy. La policy specifica quando scadrà il bucket Amazon S3. Scegli Crea per creare il bucket.

Implementa uno scenario di gioco

Note

Questo argomento si riferisce al Amazon GameLift Servers plug-in per Unity versione 1.0.0, che utilizza il server SDK 4.x o precedente.

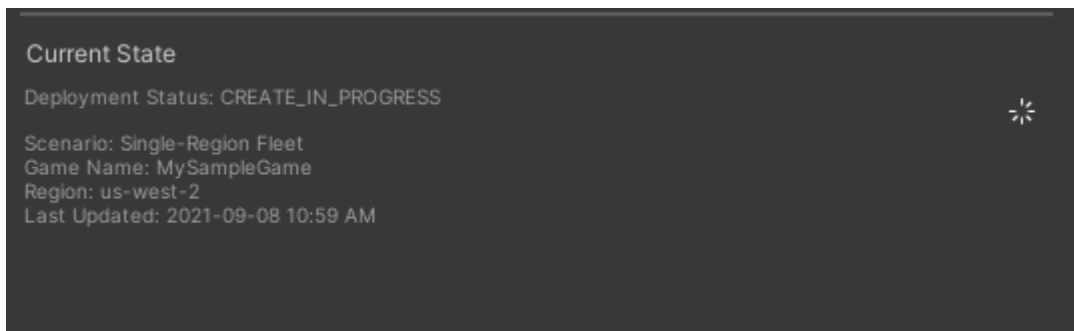
Puoi usare uno scenario per testare il tuo gioco. Amazon GameLift Servers Ogni scenario utilizza un CloudFormation modello per creare uno stack con le risorse richieste. La maggior parte degli scenari richiede un file eseguibile sul server di gioco e un percorso di compilazione. Quando distribuisce lo scenario, Amazon GameLift Servers copia le risorse di gioco nella posizione di bootstrap come parte della distribuzione.

È necessario configurare AWS le credenziali e un AWS account bootstrap per implementare uno scenario.

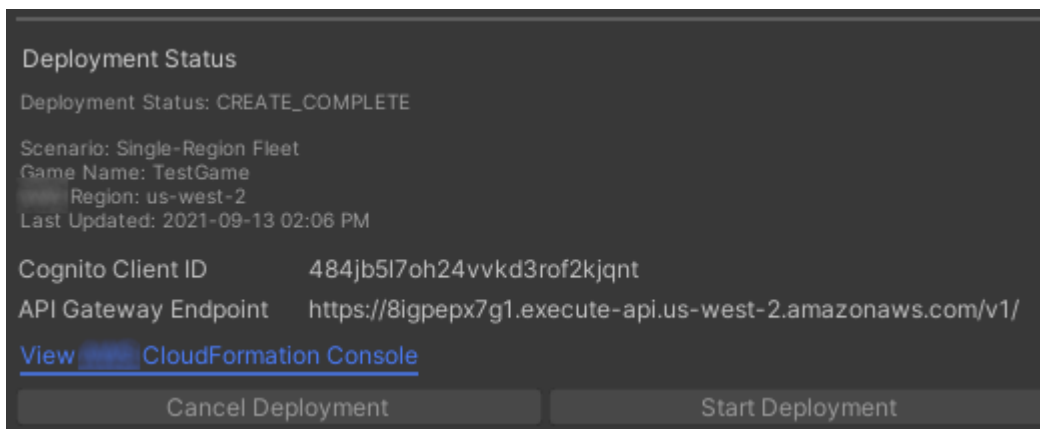
Per implementare uno scenario

1. In Unity, nel plugin per la scheda Unity, scegli la scheda Deploy.
2. Nel riquadro Deploy, scegli Open Deployment UI.
3. Nella finestra Distribuzione, scegli uno scenario.

4. Inserisci un nome per il gioco. Deve essere univoco. Il nome del gioco fa parte del nome CloudFormation dello stack quando distribuisce lo scenario.
5. Scegli il percorso della cartella di creazione del server di gioco. Il percorso della cartella di compilazione punta alla cartella contenente l'eseguibile e le dipendenze del server.
6. Scegli il percorso del file Game Server Build .exe. Il percorso del file eseguibile di build punta all'eseguibile del server di gioco.
7. Scegli Avvia distribuzione per iniziare a distribuire uno scenario. Puoi seguire lo stato dell'aggiornamento nella finestra Distribuzione in Stato corrente. La distribuzione degli scenari può richiedere diversi minuti.



8. Quando lo scenario completa la distribuzione, lo stato attuale si aggiorna per includere l'ID client Cognito e l'endpoint API Gateway che puoi copiare e incollare nel gioco.



9. Per aggiornare le impostazioni di gioco, nel menu Unity, scegli Vai alle impostazioni di connessione del client. Viene visualizzata una scheda Inspector sul lato destro della schermata Unity.
10. Deseleziona la modalità di test locale.
11. Inserite l'API Gateway Endpoint e l'ID client Cognito. Scegli lo stesso che Regione AWS hai usato per l'implementazione dello scenario. Puoi quindi ricostruire ed eseguire il client di gioco utilizzando le risorse dello scenario distribuite.

Eliminazione delle risorse create dallo scenario

Note

Questo argomento si riferisce al Amazon GameLift Servers plug-in per Unity versione 1.0.0, che utilizza il server SDK 4.x o precedente.

Per eliminare le risorse create per lo scenario, eliminate lo stack corrispondente. CloudFormation

Per eliminare le risorse create dallo scenario

1. Nella finestra del Amazon GameLift Servers plug-in per Unity Deployment, seleziona Visualizza AWS CloudFormation console per aprire la CloudFormation console.
2. Nella CloudFormation console, scegli Stacks, quindi scegli lo stack che include il nome del gioco specificato durante la distribuzione.
3. Scegli Elimina per eliminare lo stack. Potrebbero essere necessari alcuni minuti per eliminare uno stack. Dopo aver CloudFormation eliminato lo stack utilizzato dallo scenario, il suo stato diventa. ROLLBACK_COMPLETE

Integra i giochi con Unity Amazon GameLift Servers

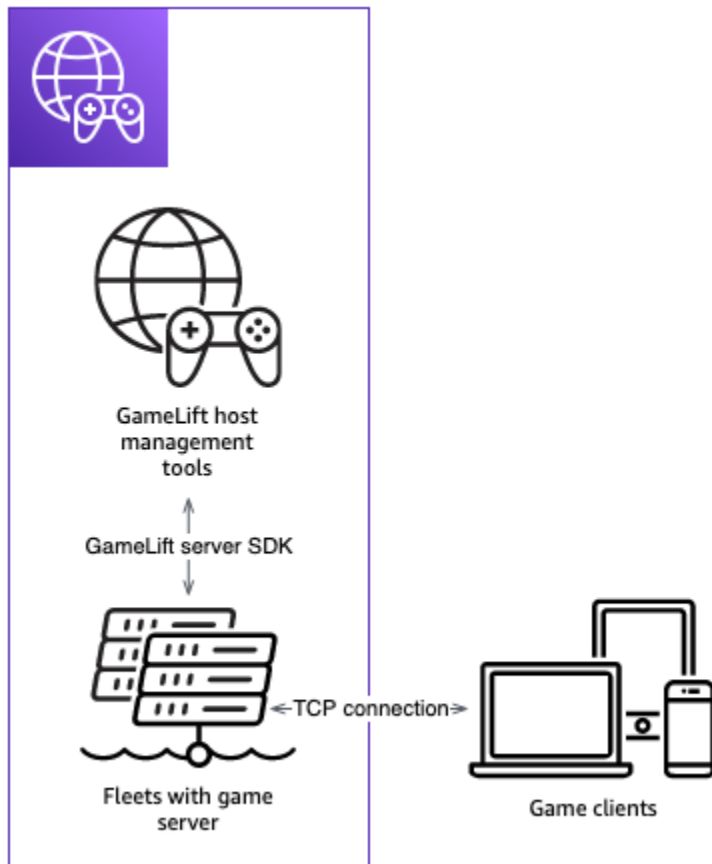
Note

Questo argomento si riferisce al Amazon GameLift Servers plug-in per Unity versione 1.0.0, che utilizza il server SDK 4.x o precedente.

Integra il tuo gioco Unity con Amazon GameLift Servers completando le seguenti attività:

- [Integrazione Amazon GameLift Servers con un progetto di server di gioco Unity](#)
- [Integrazione Amazon GameLift Servers con un progetto client di gioco Unity](#)

Il diagramma seguente mostra un esempio di flusso di integrazione di un gioco. Nel diagramma, viene schierata una flotta con il server di gioco. Amazon GameLift Servers Il client di gioco comunica con il server di gioco, che comunica con. Amazon GameLift Servers



Importa ed esegui un gioco di esempio

Note

Questo argomento si riferisce al Amazon GameLift Servers plug-in per Unity versione 1.0.0, che utilizza il server SDK 4.x o precedente.

Il Amazon GameLift Servers plugin per Unity include un gioco di esempio che puoi usare per esplorare le basi dell'integrazione con il tuo gioco. Amazon GameLift Servers In questa sezione, crei il client e il server di gioco e poi esegui i test localmente utilizzando Amazon GameLift Servers Local.

Prerequisiti

- [Configurare un account AWS utente](#)
- [Installa e configura il plugin](#)

Crea ed esegui il server di gioco di esempio

Note

Questo argomento si riferisce al Amazon GameLift Servers plug-in per la versione 1.0.0 di Unity, che utilizza il server SDK 4.x o precedente.

Configura i file del server di gioco del gioco di esempio.

1. In Unity, nel menu, scegli Amazon GameLift Servers, quindi scegli Importa gioco di esempio.
2. Nella finestra Importa gioco di esempio, scegli Importa per importare il gioco, le sue risorse e le sue dipendenze.
3. Costruisci il server di gioco. In Unity, nel menu, scegli Amazon GameLift Servers, quindi scegli Applica impostazioni build Windows Sample Server o Applica impostazioni build server di esempio macOS. Dopo aver configurato le impostazioni del server di gioco, Unity ricompila le risorse.
4. In Unity, nel menu, scegli File, quindi scegli Crea. Scegli Server Build, scegli Build, quindi scegli una cartella di build specifica per i file del server.

Unity crea il server di gioco di esempio, inserendo l'eseguibile e le risorse richieste nella cartella di build specificata.

Crea ed esegui il client di gioco di esempio

Note

Questo argomento si riferisce al Amazon GameLift Servers plug-in per la versione 1.0.0 di Unity, che utilizza il server SDK 4.x o precedente.

Configura i file client di gioco del gioco di esempio.

1. In Unity, nel menu, scegli Amazon GameLift Servers, quindi scegli Applica impostazioni di build del client di esempio di Windows o Applica impostazioni di build del client di esempio macOS. Dopo aver configurato le impostazioni del client di gioco, Unity ricompilerà le risorse.

2. In Unity, nel menu, seleziona Vai alle impostazioni del client. Verrà visualizzata una scheda Inspector sul lato destro della schermata Unity. Nella scheda Impostazioni Amazon GameLift Servers client, seleziona Local Testing Mode.
3. Crea il client di gioco. In Unity, nel menu, scegli File. Conferma che Server Build non sia selezionato, scegli Build, quindi seleziona una cartella di build specifica per i file client.

Unity crea il client di gioco di esempio, inserendo l'eseguibile e le risorse richieste nella cartella di build del client specificata.

4. Non hai creato il server e il client di gioco. Nei passaggi successivi, esegui il gioco e vedi come interagisce con Amazon GameLift Servers.

Prova il gioco di esempio a livello locale

Note

Questo argomento si riferisce al Amazon GameLift Servers plugin per Unity versione 1.0.0, che utilizza il server SDK 4.x o precedente.

Esegui il gioco di esempio che hai importato utilizzando Local. Amazon GameLift Servers

1. Avvia il server di gioco. In Unity, nel plugin per la scheda Unity, scegli la scheda Deploy.
2. Nel riquadro Test, scegli Open Local Test UI.
3. Nella finestra Local Testing, specifica il percorso del file.exe di Game Server. Il percorso deve includere il nome dell'eseguibile. Ad esempio, C:/MyGame/GameServer/MyGameServer.exe.
4. Scegli Deploy and Run. Il plug-in per Unity avvia il server di gioco e apre una finestra di registro Amazon GameLift Servers locale. La finestra contiene messaggi di registro, inclusi i messaggi inviati tra il server di gioco e Amazon GameLift Servers Local.
5. Avvia il client di gioco. Trova la posizione di costruzione con il client di gioco di esempio e scegli il file eseguibile.
6. Nel gioco Amazon GameLift Servers di esempio, fornisci un'e-mail e una password, quindi scegli Accedi. L'email e la password non vengono convalidate o utilizzate.
7. Nel gioco di Amazon GameLift Servers esempio, scegli Inizia. Il client di gioco cerca una sessione di gioco. Se non riesce a trovare una sessione, ne crea una. Il client di gioco avvia quindi la sessione di gioco. Puoi vedere l'attività di gioco nei registri.

Esempi di log del server di gioco

```
...
2021-09-15T19:55:3495 PID:20728 Log :) GAMELIFT AWAKE
2021-09-15T19:55:3512 PID:20728 Log :) I AM SERVER
2021-09-15T19:55:3514 PID:20728 Log :) GAMELIFT StartServer at port 33430.
2021-09-15T19:55:3514 PID:20728 Log :) SDK VERSION: 4.0.2
2021-09-15T19:55:3556 PID:20728 Log :) SERVER IS IN A GAMELIFT FLEET
2021-09-15T19:55:3577 PID:20728 Log :) PROCESSREADY SUCCESS.
2021-09-15T19:55:3577 PID:20728 Log :) GAMELIFT HEALTH CHECK REQUESTED (HEALTHY)
...
2021-09-15T19:55:3634 PID:20728 Log :) GAMELOGIC AWAKE
2021-09-15T19:55:3635 PID:20728 Log :) GAMELOGIC START
2021-09-15T19:55:3636 PID:20728 Log :) LISTENING ON PORT 33430
2021-09-15T19:55:3636 PID:20728 Log SERVER: Frame: 0 HELLO WORLD!
...
2021-09-15T19:56:2464 PID:20728 Log :) GAMELIFT SESSION REQUESTED
2021-09-15T19:56:2468 PID:20728 Log :) GAME SESSION ACTIVATED
2021-09-15T19:56:3578 PID:20728 Log :) GAMELIFT HEALTH CHECK REQUESTED (HEALTHY)
2021-09-15T19:57:3584 PID:20728 Log :) GAMELIFT HEALTH CHECK REQUESTED (HEALTHY)
2021-09-15T19:58:0334 PID:20728 Log SERVER: Frame: 8695 Connection accepted: playerId
  0 joined
2021-09-15T19:58:0335 PID:20728 Log SERVER: Frame: 8696 Connection accepted: playerId
  1 joined
2021-09-15T19:58:0338 PID:20728 Log SERVER: Frame: 8697 Msg rcvd from playerId 0 Msg:
  CONNECT: server IP localhost
2021-09-15T19:58:0338 PID:20728 Log SERVER: Frame: 8697 Msg rcvd from player 0:CONNECT:
  server IP localhost
2021-09-15T19:58:0339 PID:20728 Log SERVER: Frame: 8697 CONNECT: player index 0
2021-09-15T19:58:0339 PID:20728 Log SERVER: Frame: 8697 Msg rcvd from playerId 1 Msg:
  CONNECT: server IP localhost
2021-09-15T19:58:0339 PID:20728 Log SERVER: Frame: 8697 Msg rcvd from player 1:CONNECT:
  server IP localhost
2021-09-15T19:58:0339 PID:20728 Log SERVER: Frame: 8697 CONNECT: player index 1
```

Esempi di log Amazon GameLift Servers locali

```
12:55:26,000 INFO || - [SocketIOServer] main - Session store / pubsub factory used:
  MemoryStoreFactory (local session store only)
12:55:28,092 WARN || - [ServerBootstrap] main - Unknown channel option 'SO_LINGER' for
  channel '[id: 0xe23d0a14]'
12:55:28,101 INFO || - [SocketIOServer] nioEventLoopGroup-2-1 - SocketIO server
  started at port: 5757
```

```

12:55:28,101 INFO || - [SDKConnection] main - GameLift SDK server (communicates with
your game server) has started on http://localhost:5757
12:55:28,120 INFO || - [SdkWebSocketServer] WebSocketSelector-20 - WebSocket Server
started on address localhost/127.0.0.1:5759
12:55:28,166 INFO || - [StandAloneServer] main - GameLift Client server (listens for
GameLift client APIs) has started on http://localhost:8080
12:55:28,179 INFO || - [StandAloneServer] main - GameLift server sdk http listener has
started on http://localhost:5758
12:55:35,453 INFO || - [SdkWebSocketServer] WebSocketWorker-12 - onOpen
socket: /?pID=20728&sdkVersion=4.0.2&sdkLanguage=CSharp and handshake /?
pID=20728&sdkVersion=4.0.2&sdkLanguage=CSharp
12:55:35,551 INFO || - [HostProcessManager] WebSocketWorker-12 - client connected with
pID 20728
12:55:35,718 INFO || - [GameLiftSdkHttpHandler] GameLiftSdkHttpHandler-thread-0 -
GameLift API to use: ProcessReady for pId 20728
12:55:35,718 INFO || - [ProcessReadyHandler] GameLiftSdkHttpHandler-thread-0 -
Received API call for processReady from 20728
12:55:35,738 INFO || - [ProcessReadyHandler] GameLiftSdkHttpHandler-thread-0 -
onProcessReady data: port: 33430
12:55:35,739 INFO || - [HostProcessManager] GameLiftSdkHttpHandler-thread-0 -
Registered new process with pId 20728
12:55:35,789 INFO || - [GameLiftSdkHttpHandler] GameLiftSdkHttpHandler-thread-0 -
GameLift API to use: ReportHealth for pId 20728
12:55:35,790 INFO || - [ReportHealthHandler] GameLiftSdkHttpHandler-thread-0 -
Received API call for ReportHealth from 20728
12:55:35,794 INFO || - [ReportHealthHandler] GameLiftSdkHttpHandler-thread-0 -
ReportHealth data: healthStatus: true
12:56:24,098 INFO || - [GameLiftHttpHandler] Thread-12 - API to use:
GameLift.DescribeGameSessions
12:56:24,119 INFO || - [DescribeGameSessionsDispatcher] Thread-12 - Received API call
to describe game sessions with input: {"FleetId":"fleet-123"}
12:56:24,241 INFO || - [GameLiftHttpHandler] Thread-12 - API to use:
GameLift.CreateGameSession
12:56:24,242 INFO || - [CreateGameSessionDispatcher] Thread-12 - Received API call to
create game session with input: {"FleetId":"fleet-123","MaximumPlayerSessionCount":4}
12:56:24,265 INFO || - [HostProcessManager] Thread-12 - Reserved process:
20728 for gameSession: arn:aws:gamelift:local::gamesession/fleet-123/
gsess-59f6cc44-4361-42f5-95b5-fdb5825c0f3d
12:56:24,266 INFO || - [WebSocketInvoker] Thread-12 - StartGameSessionRequest:
gameSessionId=arn:aws:gamelift:local::gamesession/fleet-123/
gsess-59f6cc44-4361-42f5-95b5-fdb5825c0f3d, fleetId=fleet-123, gameSessionName=null,
maxPlayers=4, properties=[], ipAddress=127.0.0.1, port=33430, gameSessionData?=false,
matchmakerData?=false, dnsName=localhost

```

```
12:56:24,564 INFO || - [CreateGameSessionDispatcher] Thread-12 - GameSession with
id: arn:aws:gamelift:local::gamesession/fleet-123/gsess-59f6cc44-4361-42f5-95b5-
fdb5825c0f3d created
12:56:24,585 INFO || - [GameLiftHttpHandler] Thread-12 - API to use:
GameLift.DescribeGameSessions
12:56:24,585 INFO || - [DescribeGameSessionsDispatcher] Thread-12 - Received API call
to describe game sessions with input: {"FleetId":"fleet-123"}
12:56:24,660 INFO || - [GameLiftSdkHttpHandler] GameLiftSdkHttpHandler-thread-0 -
GameLift API to use: GameSessionActivate for pId 20728
12:56:24,661 INFO || - [GameSessionActivateHandler] GameLiftSdkHttpHandler-thread-0 -
Received API call for GameSessionActivate from 20728
12:56:24,678 INFO || - [GameSessionActivateHandler] GameLiftSdkHttpHandler-thread-0
- GameSessionActivate data: gameId: "arn:aws:gamelift:local::gamesession/
fleet-123/gsess-59f6cc44-4361-42f5-95b5-fdb5825c0f3d"
```

Arresta il processo del server

Note

Questo argomento si riferisce al Amazon GameLift Servers plug-in per Unity versione 1.0.0, che utilizza il server SDK 4.x o precedente.

Dopo aver finito con il gioco di esempio, spegni il server in Unity.

1. Nel client di gioco, scegli Esci o chiudi la finestra per interrompere il client di gioco.
2. In Unity, nella finestra Local Testing, scegli Stop o chiudi le finestre del server di gioco per arrestare il server.

Integra un server di gioco con Amazon GameLift Servers

Il server di gioco personalizzato deve essere in grado di interagire con esso Amazon GameLift Servers quando è distribuito e in esecuzione su risorse di hosting. Gli argomenti di questa sezione descrivono come integrare il progetto del server di gioco con l'SDK del server per Amazon GameLift Servers aggiungere funzionalità obbligatorie e opzionali. Questa sezione include istruzioni generali che puoi utilizzare con qualsiasi server specifico della lingua. SDKs Include anche una guida dettagliata sull'integrazione dei progetti di gioco Unreal Engine e Unity.

Argomenti

- [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#)
- [Integra l'SDK del server per i motori di gioco](#)

Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server

Questo argomento fornisce indicazioni generali sulla funzionalità SDK del server da aggiungere al codice del server di gioco/ Questa funzionalità consente ai processi del server di gioco di comunicare con il servizio. Amazon GameLift Servers Utilizza queste istruzioni per i server di gioco che intendi implementare su flotte EC2 Amazon GameLift Servers gestite, flotte di container gestite o flotte Anywhere.

I processi del server di gioco comunicano con il Amazon GameLift Servers servizio per ricevere istruzioni dal servizio e per segnalare lo stato dei processi del server e la sessione di gioco. Per informazioni dettagliate sulle interazioni tra i componenti della soluzione di hosting dei giochi (server di gioco, servizio di backend, client di gioco e Amazon GameLift Servers), consulta [client/server Interazioni di gioco con Amazon GameLift Servers](#).

Scarica l'SDK del server

Per preparare il gioco per l'hosting, aggiungi l'SDK del server Amazon GameLift Servers al progetto del server di gioco. Il Server SDK è disponibile in diverse lingue. Per ulteriori informazioni sul supporto degli strumenti per i server di gioco, incluso l'SDK del server, consulta. [Ottieni strumenti Amazon GameLift Servers di sviluppo](#)

[Scarica l'SDK del server per il tuo linguaggio di sviluppo da github.com/amazon-gamelift](#). Puoi trovare repository per le versioni SDK del server, i plugin e altri strumenti di sviluppo.

Note

Se stai sviluppando utilizzando Unreal Engine o Unity, scarica il Amazon GameLift Servers plug-in per questi motori di gioco. L'SDK del server è integrato e pronto all'uso. Per ulteriori informazioni, consulta questi collegamenti:

- Unreal Engine ([Scarica il plugin](#)) ([guida all'integrazione](#))
- Unity ([Scarica il plugin](#)) ([guida all'integrazione](#))

Riferimenti all'API Server SDK:

- [Server C++ SDK 5.x per -- Azioni Amazon GameLift Servers](#)
- [Server C# SDK 5.x per -- Azioni Amazon GameLift Servers](#)
- [Server C++ \(Unreal\) SDK 5.x per -- Azioni Amazon GameLift Servers](#)
- [Go server SDK per Amazon GameLift Servers -- Azioni](#)

Inizializza il processo del server

Aggiungi codice per stabilire una comunicazione con il Amazon GameLift Servers servizio e segnala quando il processo del server di gioco è pronto per ospitare una sessione di gioco. Questo codice deve essere eseguito prima di qualsiasi Amazon GameLift Servers codice.

1. Inizializza un client Amazon GameLift Servers API `InitSdk()` chiamando. [Se stai preparando il tuo server di gioco per l'esecuzione su flotte EC2 Amazon GameLift Servers gestite, usa l'impostazione predefinita `InitSDK\(\)` \(C++\) \(C#\) \(Unreal\) \(Go\) \(C++\) senza parametri.](#) Il client API gestisce la connessione al servizio per te. Amazon GameLift Servers

 Se stai preparando il tuo server di gioco per l'uso su una flotta Amazon GameLift Servers Anywhere:

Inizializza il client Amazon GameLift Servers API chiamando `InitSdk()` con quanto segue: `ServerParameters`

- L'URL del websocket utilizzato per connettersi al server di gioco.
- L'ID del processo utilizzato per ospitare il server di gioco.
- L'ID del computer che ospita i processi del server di gioco.
- L'ID della flotta contenente il tuo computer Amazon GameLift Servers Anywhere.
- Il token di autorizzazione generato dall'Amazon GameLift Servers operazione.

[GetComputeAuthToken](#)

2. Notifica al servizio che il processo del server di gioco è pronto per ospitare una sessione di gioco. Chiama `ProcessReady()` ([C++](#)) ([C#](#)) ([Unreal](#)) ([Go](#)) (C++) con quanto segue. `ProcessParameters` Ogni processo del server di gioco deve chiamare solo una volta. `ProcessReady()`

- Il numero di porta per il processo server. Quando il processo server avvia una sessione di gioco, fornisce la porta al Amazon GameLift Servers servizio, che aggiorna le informazioni

sulla sessione di gioco. Il gioco può recuperare queste informazioni e fornirle ai client di gioco, che le utilizzano per connettersi al processo del server e partecipare alla sessione di gioco.

- Le posizioni dei file che desideri Amazon GameLift Servers archiviare per te. Questi potrebbero includere i registri delle sessioni di gioco e altri file generati dal processo del server durante una sessione di gioco. Sebbene Amazon GameLift Servers salvi temporaneamente questi file sul computer in cui è in esecuzione il processo del server, questi file sono disponibili solo fino alla chiusura dell'istanza. È possibile accedere ai file archiviati tramite la [Amazon GameLift Servers console](#) o chiamando l'operazione Amazon GameLift Servers API [GetGameSessionLogUrl\(\)](#).

 Se stai preparando il tuo server di gioco per l'uso su una flotta di container Amazon GameLift Servers gestita:

Non è necessario specificare i parametri di registro per una flotta di container. Invia invece la sessione di gioco e altri dati di registro allo standard output. Le flotte di container acquisiscono automaticamente tutti gli output standard dei container come flusso di log.

- Le seguenti funzioni di callback che consentono di Amazon GameLift Servers inviare messaggi o prompt a un processo del server di gioco. È necessario implementare ognuna di queste funzioni nel codice del server di gioco. [Per ulteriori informazioni, consulta `ProcessParameters\(C++\)` \(`C#`\) \(`Unreal`\) \(`Go`\) \(`C++Unreal`\).](#)
- (Facoltativo)`onHealthCheck`: Amazon GameLift Servers chiama questa funzione regolarmente per richiedere un rapporto sullo stato di salute del server.
- `onStartGameSession`— Amazon GameLift Servers chiama questa funzione in risposta alla richiesta del client [CreateGameSession\(\)](#).
- `onProcessTerminate`— Amazon GameLift Servers forza l'arresto del processo del server, lasciandolo spegnersi correttamente.
- (Facoltativo)`onUpdateGameSession`: Amazon GameLift Servers fornisce un oggetto della sessione di gioco aggiornato al server di gioco o fornisce un aggiornamento dello stato su una richiesta di ripristino della partita. La funzione di [FlexMatchriempimento](#) richiede questo callback.

Puoi anche configurare un server di gioco in modo che possa accedere in modo sicuro ad altre AWS risorse che possiedi o controlli. Per ulteriori informazioni, consulta [Connect il server di gioco Amazon GameLift Servers ospitato ad altre AWS risorse](#).

(Facoltativo) Segnala lo stato dei processi del server

Aggiungi codice al tuo server di gioco per implementare la funzione `onHealthCheck()` di callback. Amazon GameLift Servers richiama periodicamente questo metodo di callback per raccogliere metriche sullo stato di salute. Per implementare questa funzione di callback, procedi come segue:

- Valuta lo stato di salute del processo del server. Ad esempio, è possibile segnalare che il processo del server non è integro se alcune dipendenze esterne non sono riuscite.
- Completare la valutazione dello stato e rispondere alla richiamata entro 60 secondi. Se Amazon GameLift Servers non riceve una risposta entro tale lasso di tempo, considera automaticamente il processo del server non integro.
- Restituisce un valore booleano: `true` per salutare, `false` per non salutare.

Se non implementi un callback per il controllo dello stato di salute, Amazon GameLift Servers considera il processo del server integro a meno che il server non risponda.

Il Amazon GameLift Servers servizio utilizza l'integrità dei processi del server per porre fine a processi non integri e ripulire le risorse. Se un processo del server continua a essere segnalato come non integro o non risponde a tre controlli di integrità consecutivi, il servizio potrebbe arrestare il processo e avviarne uno nuovo. Il servizio raccoglie le metriche sullo stato dei processi del server di un parco macchine.

(Facoltativo) Ottieni un certificato TLS

Se il processo server è in esecuzione su una flotta in cui è attivata la generazione di certificati TLS, puoi recuperare il certificato TLS per stabilire una connessione sicura con un client di gioco e crittografare le comunicazioni client-server. Una copia del certificato viene archiviata nell'istanza. [Per conoscere la posizione del file, chiamate `GetComputeCertificate\(\)` \(C++\) \(C#\) \(Unreal\) \(Go\) \(C++\) \(C#\) \(Unreal\).](#)

Inizia una sessione di gioco

Aggiungi codice per implementare la funzione di callback. `onStartGameSession` Amazon GameLift Servers richiama questo callback per avviare una sessione di gioco sul processo del server.

La `onStartGameSession` funzione accetta un [GameSession](#) oggetto come parametro di input. Questo oggetto include informazioni chiave sulla sessione di gioco, come il numero massimo di giocatori. Può anche includere dati di gioco e dati dei giocatori. L'implementazione della funzione dovrebbe svolgere le seguenti attività:

- Avvia azioni per creare una nuova sessione di gioco in base alle `GameSession` proprietà. Come minimo, il server di gioco deve associare l'ID della sessione di gioco, a cui fanno riferimento i client di gioco quando si connettono al processo del server.
- Elabora i dati di gioco e i dati dei giocatori secondo necessità. Questi dati si trovano nell'`GameSession` oggetto.
- Avvisa il Amazon GameLift Servers servizio quando una nuova sessione di gioco è pronta per accettare giocatori. [Chiama l'operazione dell'API del server `ActivateGameSession\(\)` \(C++\) \(C#\) \(Unreal\) \(Go\) \(C++\) Unreal](#). In risposta a una chiamata riuscita, il servizio modifica lo stato della sessione di gioco in. `ACTIVE`

(Facoltativo) Convalida un nuovo giocatore

Se stai monitorando lo stato delle sessioni dei giocatori, aggiungi il codice per convalidare un nuovo giocatore quando si connette a un server di gioco. Amazon GameLift Servers tiene traccia dei giocatori attuali e degli slot disponibili per le sessioni di gioco.

Per la convalida, un client di gioco che tenta di partecipare a una sessione di gioco deve includere un ID di sessione del giocatore. Amazon GameLift Servers genera questo ID quando il gioco inizia nuove sessioni di gioco chiamando [StartGameSessionPlacement\(\)](#) o [StartMatchmaking\(\)](#). Su queste richieste, uno slot libero in una sessione di gioco è riservato alla sessione del giocatore.

Quando il processo del server di gioco riceve una richiesta di connessione al client di gioco, chiama `AcceptPlayerSession()` ([C++](#)) ([C#](#)) ([Unreal](#)) ([Go](#)) ([C++](#)) della sessione del giocatore. In risposta, Amazon GameLift Servers verifica che l'ID della sessione del giocatore corrisponda a uno slot aperto riservato nella sessione di gioco. Dopo aver Amazon GameLift Servers convalidato l'ID della sessione del giocatore, il processo del server accetta la connessione. Il giocatore può quindi partecipare alla sessione di gioco. Se Amazon GameLift Servers non convalida l'ID della sessione del giocatore, il processo del server nega la connessione.

(Facoltativo) Segnala la fine di una sessione di gioco

Se stai monitorando lo stato delle sessioni dei giocatori, aggiungi il codice per avvisare Amazon GameLift Servers quando un giocatore abbandona la sessione di gioco. Questo codice dovrebbe essere eseguito ogni volta che il processo del server rileva una connessione interrotta. Amazon GameLift Servers utilizza questa notifica per tenere traccia dei giocatori attuali e degli slot disponibili nella sessione di gioco.

Per gestire le connessioni interrotte nel codice, aggiungi una chiamata all'operazione dell'API del server `RemovePlayerSession()` ([C++](#)) ([C#](#)) ([Unreal](#)) ([Go](#)) (C++) di sessione del giocatore corrispondente.

Termina una sessione di gioco

Aggiungi codice alla sequenza di chiusura del processo del server per notificare Amazon GameLift Servers quando una sessione di gioco sta per terminare. Per riciclare e aggiornare le risorse di hosting, chiudi ogni processo del server al termine della sessione di gioco.

[All'avvio del codice di spegnimento del processo server, richiama l'operazione dell'API del server `ProcessEnding\(\)` \(C++\) \(C#\) \(Unreal\) \(Go\) g \(C++\) \(C#\)Unreal](#). Questa chiamata notifica Amazon GameLift Servers che il processo del server si sta chiudendo. Amazon GameLift Servers modifica lo stato della sessione di gioco e lo stato del processo del server in `TERMINATED`. Dopo la chiamata `ProcessEnding()`, è sicuro che il processo si interrompa.

Rispondi a una notifica di chiusura del processo del server

Aggiungi codice per arrestare il processo del server in risposta a una notifica del Amazon GameLift Servers servizio. Il servizio invia questa notifica quando il processo del server segnala costantemente che non è integro o se l'istanza in cui è in esecuzione il processo server viene interrotta. Amazon GameLift Servers può arrestare un'istanza come parte di un evento di riduzione della capacità o in risposta all'interruzione dell'istanza Spot. Un'interruzione di un'istanza Spot fornisce un preavviso di due minuti, che dà al server il tempo necessario per disconnettere correttamente i giocatori, conservare i dati sullo stato del gioco ed eseguire altre attività di pulizia.

Per gestire una notifica di spegnimento, apporta le seguenti modifiche al codice del server di gioco:

- [Implementa la funzione di callback `onProcessTerminate\(\)` \(C++\) \(C#\) \(Unreal\) \(Go\) \(C++\) \(C#\)](#). Questa funzione dovrebbe richiamare il codice che arresta il processo del server.
- Richiama l'operazione dell'API del server `GetTerminationTime()` ([C++](#)) ([C#](#)) ([Unreal](#)) ([Go](#)) (C++) dal codice di spegnimento del server di gioco. Se Amazon GameLift Servers ha emesso una chiamata per interrompere il processo del server, restituisce il tempo di terminazione stimato. `GetTerminationTime()`
- [All'inizio del codice di spegnimento del server di gioco, richiama l'operazione dell'API del server `ProcessEnding\(\)` \(C++\) \(C#\) \(Unreal\) \(Go\) \(C++\) \(C#\)](#). Questa chiamata avvisa il servizio Amazon GameLift Servers che il processo del server è in fase di arresto. Il servizio modifica quindi lo stato del processo del server in `TERMINATED`. Dopo la chiamata `ProcessEnding()`, è sicuro che il processo si interrompa.

Integra l'SDK del server per i motori di gioco

Puoi utilizzare il Amazon GameLift Servers servizio gestito con la maggior parte dei principali motori di gioco che supportano le librerie C++ o C#, tra cui O3DE, Unreal Engine e Unity. Crea la versione di cui hai bisogno per il gioco; consulta i file README di ciascuna versione per le istruzioni per la compilazione e i requisiti minimi. Per ulteriori informazioni sulle piattaforme di sviluppo e sui sistemi operativi disponibili e supportati Amazon GameLift Servers SDKs, consulta la sezione dedicata ai server di gioco. [Ottieni strumenti Amazon GameLift Servers di sviluppo](#)

Oltre alle informazioni specifiche del motore fornite in questo argomento, negli argomenti seguenti è possibile trovare ulteriore assistenza sull'integrazione di Amazon GameLift Servers nei server di gioco, nei servizi e nei client:

- [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#): istruzioni dettagliate sull'integrazione di Amazon GameLift Servers in un server di gioco.
- [Integra la funzionalità del client di Amazon GameLift Servers gioco](#): istruzioni dettagliate sull'integrazione in un servizio o client di gioco, tra cui la creazione di sessioni di gioco e l'aggiunta di utenti ai giochi.

Unreal Engine

Server di gioco

Prepara i server di gioco per l'hosting Amazon GameLift Servers aggiungendo l'[SDK del server Amazon GameLift Servers per Unreal Engine](#) al tuo progetto e implementando le funzionalità del server richieste. Per assistenza nella configurazione del plug-in Unreal Engine e l'aggiunta di codice Amazon GameLift Servers, consulta [Integrazione Amazon GameLift Servers in un progetto Unreal Engine](#).

Client e servizi di gioco

Consenti ai servizi di and/or gioco dei tuoi client di gioco di interagire con il Amazon GameLift Servers servizio, ad esempio per trovare sessioni di gioco disponibili o crearne di nuove e aggiungere giocatori ai giochi. Le funzionalità client di base sono fornite nell'[AWS SDK for C++](#). Per integrare Amazon GameLift Servers nel progetto di gioco Unreal Engine, consulta [Integra la funzionalità del client di Amazon GameLift Servers gioco](#).

Unity

Server di gioco

Prepara i server di gioco per l'hosting Amazon GameLift Servers aggiungendo il [server SDK Amazon GameLift Servers for C#](#) al tuo progetto e implementando le funzionalità server richieste. Per assistenza nella configurazione con Unity e l'aggiunta di codice Amazon GameLift Servers, consulta [Integrazione Amazon GameLift Servers in un progetto Unity](#).

Client e servizi di gioco

Consenti ai servizi di and/or gioco dei tuoi client di gioco di interagire con il Amazon GameLift Servers servizio, ad esempio per trovare sessioni di gioco disponibili o crearne di nuove e aggiungere giocatori ai giochi. La funzionalità client principale è fornita da [AWS SDK per .NET](#). Per integrare Amazon GameLift Servers nel progetto di gioco Unity, consulta [Integra la funzionalità del client di Amazon GameLift Servers gioco](#).

Altri motori

Per un elenco completo dei server e dei client di gioco Amazon GameLift Servers SDKs disponibili, vedi [the section called “Ottieni strumenti di sviluppo”](#).

Integrazione Amazon GameLift Servers in un progetto Unreal Engine

Scopri come integrare l'Amazon GameLift Servers SDK per Unreal Engine nei tuoi progetti di gioco per accedere al set completo di funzionalità SDK del server.

Suggerimento

Per una distribuzione più rapida, prova il plug-in Amazon GameLift Servers standalone per Unreal Engine. Fornisce flussi di lavoro guidati per l'interfaccia utente per implementare rapidamente il server di gioco con una configurazione minima, in modo da poter provare i componenti del gioco in azione. Per informazioni, consulta [Amazon GameLift Servers plugin per Unreal Engine](#).

Risorse aggiuntive:

- [Server C++ \(Unreal\) SDK 5.x per -- Azioni Amazon GameLift Servers](#)
- [Ottieni strumenti Amazon GameLift Servers di sviluppo](#)

Installa l'SDK del server per Unreal

Scarica l'Amazon GameLift ServersSDK open source per Unreal Engine da [GitHub](#) I file readme del repository contengono i prerequisiti e le istruzioni di installazione.

Imposta gli obiettivi di compilazione e le regole del modulo

Modifica i file di progetto del gioco per generare correttamente i componenti di compilazione da utilizzare con Amazon GameLift Servers.

Per aggiungere obiettivi di build per client e server:

1. Apri i file di codice del progetto di gioco e individua il file del `.../Games/[your application name]Source/[your application name]Target.cs` file. Esempio: `.../Source/GameLiftUnrealAppTarget.cs`. (Se usi Visual Studio, apri il `.sln` file del progetto).
2. Copia questo file per creare due nuovi file di destinazione nella `Source/` directory.
 - Target client: rinomina il nuovo file in `[your application name]Client.Target.cs`. Modifica il contenuto per aggiornare i valori del nome della classe e del tipo di destinazione, come illustrato nel seguente codice di esempio:

```
using UnrealBuildTool;
using System.Collections.Generic;

public class GameLiftUnrealAppClientTarget : TargetRules
{
    public GameLiftUnrealAppClientTarget ( TargetInfo Target ) : base
    ( Target )
    {
        Type = TargetType.Client;
        DefaultBuildSettings = BuildSettingsVersion.V2;
        IncludeOrderVersion = EngineIncludeOrderVersion.Unreal5_1;
        ExtraModuleNames.Add( "GameLiftUnrealApp" );
    }
}
```

- Destinazione del server: rinomina il nuovo file in `[your application name]Server.Target.cs` Modifica il contenuto per aggiornare i valori del nome della classe e del tipo di destinazione, come illustrato nel seguente codice di esempio:

```
using UnrealBuildTool;
using System.Collections.Generic;
```

```

public class GameLiftUnrealAppServerTarget : TargetRules
{
    public GameLiftUnrealAppServerTarget ( TargetInfo Target ) : base
    ( Target )
    {
        Type = TargetType.Server;
        DefaultBuildSettings = BuildSettingsVersion.V2;
        IncludeOrderVersion = EngineIncludeOrderVersion.Unreal5_1;
        ExtraModuleNames.Add( "GameLiftUnrealApp");
    }
}

```

3. Rigenera i file di progetto. Se usi Visual Studio, puoi fare clic con il pulsante destro del mouse sul .uproject file del progetto di gioco e selezionare Genera file di progetto di Visual Studio.

Per aggiornare le regole del modulo del progetto di gioco:

Aggiorna le regole del modulo del progetto di gioco per fare in modo che dipenda dal plugin.

1. Apri i file di codice del progetto di gioco e individua il .../Games/[*your application name*]Source/[*your application name*].Build.cs file. Esempio: .../Source/GameLiftUnrealApp.Build.cs. (Se usi Visual Studio, apri il .sln file del progetto).
2. Individua la ModuleRules classe e aggiorna come illustrato nel seguente codice di esempio:

```

using UnrealBuildTool;

public class GameLiftUnrealApp : ModuleRules
{
    public GameLiftUnrealApp ( ReadOnlyTargetRules Target ) : base ( Target )
    {
        PCHUsage = PCHUsageMode.UseExplicitOrSharedPCHs;
        PublicDependencyModuleNames.AddRange( new string[] { "Core",
"CoreUObject", "Engine", "InputCore", "HeadMountedDisplay",
"EnhancedInput" });
        // Add the following section
        if (Target.Type == TargetType.Server)
        {
            PublicDependencyModuleNames.Add("GameLiftServerSDK");
        }
        else
        {

```

```
        PublicDefinitions.Add("WITH_GAMELIFT=0");
    }
    bEnableExceptions = true;
}
}
```

3. Dopo aver creato i nuovi file di destinazione e modificato le regole del modulo, ricostruisci il progetto del gioco.

Aggiungi funzionalità di hosting di giochi al codice del tuo server

Dopo l'installazione e la configurazione dell'SDK del server, il passaggio successivo consiste nell'integrare la funzionalità di hosting dei giochi nel codice del server. L'SDK del server consente al server di gioco di comunicare con il Amazon GameLift Servers servizio, ricevere istruzioni per le sessioni di gioco, segnalare lo stato e lo stato di salute ed eseguire altre azioni.

Questo argomento fornisce un esempio di codice che aggiunge le funzionalità minime necessarie per ospitare il gioco. Amazon GameLift Servers

Passaggio 1: Aggiornare il file di **GameMode** intestazione

1. Apri i file di codice del progetto di gioco e individua il *Your-application-name*GameMode.h file. Esempio: GameLiftUnrealAppGameMode.h. Se usi Visual Studio, apri il .sln file relativo al progetto del gioco.
2. Modifica il file di intestazione per includere il seguente codice di esempio. Assicurati di sostituire "GameLiftUnrealApp" con il nome della tua applicazione.

Codice di esempio per GameMode.h

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0

#pragma once

#include "CoreMinimal.h"
#include "GameFramework/GameModeBase.h"
#include "GameLiftUnrealAppGameMode.generated.h"

struct FProcessParameters;
```

```

DECLARE_LOG_CATEGORY_EXTERN(GameServerLog, Log, All);

UCLASS(minimalapi)
class AGameLiftUnrealAppGameMode : public AGameModeBase
{
    GENERATED_BODY()

public:
    AGameLiftUnrealAppGameMode();

protected:
    virtual void BeginPlay() override;

private:
    void InitGameLift();

private:
    TSharedPtr<FProcessParameters> ProcessParameters;
};

```

Passaggio 2: aggiungi le chiamate SDK del server richieste al codice del server di gioco

Usa il codice di esempio in questa sezione per integrare il codice del server di gioco da utilizzare con Amazon GameLift Servers. Per informazioni dettagliate sulle funzioni del codice, consulta [Inizializza il processo del server](#) e [Server C++ \(Unreal\) SDK 5.x per -- Azioni Amazon GameLift Servers](#).

Note

Il flag `WITH_GAMELIFT` del preprocessore ha due scopi:

- Limita le chiamate API di Amazon GameLift Servers backend solo alle build del server Unreal
- Garantisce la compatibilità tra i diversi obiettivi di build di Unreal

1. Aprire il ***Your-application-name***GameMode.cpp file sorgente correlato. Nel nostro esempio:GameLiftUnrealAppGameMode.cpp.
2. Modifica il codice per allinearli al seguente codice di esempio. Assicuratevi di sostituire qualsiasi istanza di "GameLiftUnrealApp" con il nome della vostra applicazione.

L'esempio di codice fornito mostra come aggiungere gli elementi richiesti per l'integrazione con Amazon GameLift Servers. Ciò include:

- Inizializza un client Amazon GameLift Servers API.
- Implementa funzioni di callback per rispondere alle richieste del Amazon GameLift Servers servizio, tra cui `OnStartGameSessionOnProcessTerminate`, e. `onHealthCheck`
- Chiama `ProcessReady()` per avvisare il Amazon GameLift Servers servizio quando è pronto per ospitare sessioni di gioco.

Codice di esempio per `GameMode.cpp`

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0

#include "GameLiftUnrealAppGameMode.h"

#include "UObject/ConstructorHelpers.h"
#include "Kismet/GameplayStatics.h"

#if WITH_GAMELIFT
#include "GameLiftServerSDK.h"
#include "GameLiftServerSDKModels.h"
#endif

#include "GenericPlatform/GenericPlatformOutputDevices.h"

DEFINE_LOG_CATEGORY(GameServerLog);

AGameLiftUnrealAppGameMode::AGameLiftUnrealAppGameMode() :
    ProcessParameters(nullptr)
{
    // Set default pawn class to our Blueprinted character
    static ConstructorHelpers::FClassFinder<APawn> PlayerPawnBPClass(TEXT("/Game/ThirdPerson/Blueprints/BP_ThirdPersonCharacter"));

    if (PlayerPawnBPClass.Class != NULL)
    {
        DefaultPawnClass = PlayerPawnBPClass.Class;
    }
}
```

```

    UE_LOG(GameServerLog, Log, TEXT("Initializing AGameLiftUnrealAppGameMode..."));
}

void AGameLiftUnrealAppGameMode::BeginPlay()
{
    Super::BeginPlay();

#ifdef WITH_GAMELIFT
    InitGameLift();
#endif
}

void AGameLiftUnrealAppGameMode::InitGameLift()
{
#ifdef WITH_GAMELIFT
    UE_LOG(GameServerLog, Log, TEXT("Calling InitGameLift..."));

    // Getting the module first.
    FGameLiftServerSDKModule* GameLiftSdkModule =
    &FModuleManager::LoadModuleChecked<FGameLiftServerSDKModule>(FName("GameLiftServerSDK"));

    //Define the server parameters for a GameLift Anywhere fleet. These are not needed
    for a GameLift managed EC2 fleet.
    FServerParameters ServerParametersForAnywhere;

    bool bIsAnywhereActive = false;
    if (FParse::Param(FCommandLine::Get(), TEXT("glAnywhere")))
    {
        bIsAnywhereActive = true;
    }

    if (bIsAnywhereActive)
    {
        UE_LOG(GameServerLog, Log, TEXT("Configuring server parameters for
Anywhere..."));

        // If GameLift Anywhere is enabled, parse command line arguments and pass them
        in the ServerParameters object.
        FString glAnywhereWebSocketUrl = "";
        if (FParse::Value(FCommandLine::Get(), TEXT("glAnywhereWebSocketUrl="),
glAnywhereWebSocketUrl))
        {
            ServerParametersForAnywhere.m_webSocketUrl =
TCHAR_TO_UTF8(*glAnywhereWebSocketUrl);

```

```

    }

    FString glAnywhereFleetId = "";
    if (FParse::Value(FCommandLine::Get(), TEXT("glAnywhereFleetId="),
glAnywhereFleetId))
    {
        ServerParametersForAnywhere.m_fleetId = TCHAR_TO_UTF8(*glAnywhereFleetId);
    }

    FString glAnywhereProcessId = "";
    if (FParse::Value(FCommandLine::Get(), TEXT("glAnywhereProcessId="),
glAnywhereProcessId))
    {
        ServerParametersForAnywhere.m_processId =
TCHAR_TO_UTF8(*glAnywhereProcessId);
    }
    else
    {
        // If no ProcessId is passed as a command line argument, generate a
randomized unique string.
        FString TimeString = FString::FromInt(std::time(nullptr));
        FString ProcessId = "ProcessId_" + TimeString;
        ServerParametersForAnywhere.m_processId = TCHAR_TO_UTF8(*ProcessId);
    }

    FString glAnywhereHostId = "";
    if (FParse::Value(FCommandLine::Get(), TEXT("glAnywhereHostId="),
glAnywhereHostId))
    {
        ServerParametersForAnywhere.m_hostId = TCHAR_TO_UTF8(*glAnywhereHostId);
    }

    FString glAnywhereAuthToken = "";
    if (FParse::Value(FCommandLine::Get(), TEXT("glAnywhereAuthToken="),
glAnywhereAuthToken))
    {
        ServerParametersForAnywhere.m_authToken =
TCHAR_TO_UTF8(*glAnywhereAuthToken);
    }

    FString glAnywhereAwsRegion = "";
    if (FParse::Value(FCommandLine::Get(), TEXT("glAnywhereAwsRegion="),
glAnywhereAwsRegion))
    {

```

```

        ServerParametersForAnywhere.m_awsRegion =
TCHAR_TO_UTF8(*glAnywhereAwsRegion);
    }

    FString glAnywhereAccessKey = "";
    if (FParse::Value(FCommandLine::Get(), TEXT("glAnywhereAccessKey="),
glAnywhereAccessKey))
    {
        ServerParametersForAnywhere.m_accessKey =
TCHAR_TO_UTF8(*glAnywhereAccessKey);
    }

    FString glAnywhereSecretKey = "";
    if (FParse::Value(FCommandLine::Get(), TEXT("glAnywhereSecretKey="),
glAnywhereSecretKey))
    {
        ServerParametersForAnywhere.m_secretKey =
TCHAR_TO_UTF8(*glAnywhereSecretKey);
    }

    FString glAnywhereSessionToken = "";
    if (FParse::Value(FCommandLine::Get(), TEXT("glAnywhereSessionToken="),
glAnywhereSessionToken))
    {
        ServerParametersForAnywhere.m_sessionToken =
TCHAR_TO_UTF8(*glAnywhereSessionToken);
    }

    UE_LOG(GameServerLog, SetColor, TEXT("%s"), COLOR_YELLOW);
    UE_LOG(GameServerLog, Log, TEXT(">>>> WebSocket URL: %s"),
*ServerParametersForAnywhere.m_webSocketUrl);
    UE_LOG(GameServerLog, Log, TEXT(">>>> Fleet ID: %s"),
*ServerParametersForAnywhere.m_fleetId);
    UE_LOG(GameServerLog, Log, TEXT(">>>> Process ID: %s"),
*ServerParametersForAnywhere.m_processId);
    UE_LOG(GameServerLog, Log, TEXT(">>>> Host ID (Compute Name): %s"),
*ServerParametersForAnywhere.m_hostId);
    UE_LOG(GameServerLog, Log, TEXT(">>>> Auth Token: %s"),
*ServerParametersForAnywhere.m_authToken);
    UE_LOG(GameServerLog, Log, TEXT(">>>> Aws Region: %s"),
*ServerParametersForAnywhere.m_awsRegion);
    UE_LOG(GameServerLog, Log, TEXT(">>>> Access Key: %s"),
*ServerParametersForAnywhere.m_accessKey);

```



```

        UE_LOG(GameServerLog, Log, TEXT(">>>> Secret Key: %s"),
*ServerParametersForAnywhere.m_secretKey);
        UE_LOG(GameServerLog, Log, TEXT(">>>> Session Token: %s"),
*ServerParametersForAnywhere.m_sessionToken);
        UE_LOG(GameServerLog, SetColor, TEXT("%s"), COLOR_NONE);
    }

    UE_LOG(GameServerLog, Log, TEXT("Initializing the GameLift Server..."));

    //InitSDK will establish a local connection with GameLift's agent to enable further
communication.
    FGameLiftGenericOutcome InitSdkOutcome = GameLiftSdkModule-
>InitSDK(ServerParametersForAnywhere);
    if (InitSdkOutcome.IsSuccess())
    {
        UE_LOG(GameServerLog, SetColor, TEXT("%s"), COLOR_GREEN);
        UE_LOG(GameServerLog, Log, TEXT("GameLift InitSDK succeeded!"));
        UE_LOG(GameServerLog, SetColor, TEXT("%s"), COLOR_NONE);
    }
    else
    {
        UE_LOG(GameServerLog, SetColor, TEXT("%s"), COLOR_RED);
        UE_LOG(GameServerLog, Log, TEXT("ERROR: InitSDK failed : ("));
        FGameLiftError GameLiftError = InitSdkOutcome.GetError();
        UE_LOG(GameServerLog, Log, TEXT("ERROR: %s"), *GameLiftError.m_errorMessage);
        UE_LOG(GameServerLog, SetColor, TEXT("%s"), COLOR_NONE);
        return;
    }

    ProcessParameters = MakeShared<FProcessParameters>();

    //When a game session is created, Amazon GameLift Servers sends an activation
request to the game server and passes along the game session object containing game
properties and other settings.
    //Here is where a game server should take action based on the game session object.
    //Once the game server is ready to receive incoming player connections, it should
invoke GameLiftServerAPI.ActivateGameSession()
    ProcessParameters->OnStartGameSession.BindLambda([=]
(Aws::GameLift::Server::Model::GameSession InGameSession)
    {
        FString GameSessionId = FString(InGameSession.GetGameSessionId());
        UE_LOG(GameServerLog, Log, TEXT("GameSession Initializing: %s"),
*GameSessionId);
        GameLiftSdkModule->ActivateGameSession();
    });

```

```

});

//OnProcessTerminate callback. Amazon GameLift Servers will invoke this callback
before shutting down an instance hosting this game server.
//It gives this game server a chance to save its state, communicate with services,
etc., before being shut down.
//In this case, we simply tell Amazon GameLift Servers we are indeed going to shut
down.
ProcessParameters->OnTerminate.BindLambda([=]()
{
    UE_LOG(GameServerLog, Log, TEXT("Game Server Process is terminating"));
    // First call ProcessEnding()
    FGameLiftGenericOutcome processEndingOutcome = GameLiftSdkModule-
>ProcessEnding();
    // Then call Destroy() to free the SDK from memory
    FGameLiftGenericOutcome destroyOutcome = GameLiftSdkModule->Destroy();
    // Exit the process with success or failure
    if (processEndingOutcome.IsSuccess() && destroyOutcome.IsSuccess()) {
        UE_LOG(GameServerLog, Log, TEXT("Server process ending successfully"));
    }
    else {
        if (!processEndingOutcome.IsSuccess()) {
            const FGameLiftError& error = processEndingOutcome.GetError();
            UE_LOG(GameServerLog, Error, TEXT("ProcessEnding() failed. Error:
%s"),
                error.m_errorMessage.IsEmpty() ? TEXT("Unknown error") :
*error.m_errorMessage);
        }
        if (!destroyOutcome.IsSuccess()) {
            const FGameLiftError& error = destroyOutcome.GetError();
            UE_LOG(GameServerLog, Error, TEXT("Destroy() failed. Error: %s"),
                error.m_errorMessage.IsEmpty() ? TEXT("Unknown error") :
*error.m_errorMessage);
        }
    }
});

//This is the HealthCheck callback.
//Amazon GameLift Servers will invoke this callback every 60 seconds or so.
//Here, a game server might want to check the health of dependencies and such.
//Simply return true if healthy, false otherwise.
//The game server has 60 seconds to respond with its health status. Amazon GameLift
Servers will default to 'false' if the game server doesn't respond in time.
//In this case, we're always healthy!

```

```

ProcessParameters->OnHealthCheck.BindLambda([]()
{
    UE_LOG(GameServerLog, Log, TEXT("Performing Health Check"));
    return true;
});

//GameServer.exe -port=7777 LOG=server.mylog
ProcessParameters->port = FURL::UrlConfig.DefaultPort;
TArray<FString> CommandLineTokens;
TArray<FString> CommandLineSwitches;

FCommandLine::Parse(FCommandLine::Get(), CommandLineTokens, CommandLineSwitches);

for (FString SwitchStr : CommandLineSwitches)
{
    FString Key;
    FString Value;

    if (SwitchStr.Split("=", &Key, &Value))
    {
        if (Key.Equals("port"))
        {
            ProcessParameters->port = FString::Atoi(*Value);
        }
    }
}

//Here, the game server tells Amazon GameLift Servers where to find game session
log files.
//At the end of a game session, Amazon GameLift Servers uploads everything in the
specified
//location and stores it in the cloud for access later.
TArray<FString> Logfiles;
Logfiles.Add(TEXT("GameLiftUnrealApp/Saved/Logs/server.log"));
ProcessParameters->logParameters = Logfiles;

//The game server calls ProcessReady() to tell Amazon GameLift Servers it's ready
to host game sessions.
UE_LOG(GameServerLog, Log, TEXT("Calling Process Ready..."));
FGameLiftGenericOutcome ProcessReadyOutcome = GameLiftSdkModule-
>ProcessReady(*ProcessParameters);

if (ProcessReadyOutcome.IsSuccess())
{

```

```
        UE_LOG(GameServerLog, SetColor, TEXT("%s"), COLOR_GREEN);
        UE_LOG(GameServerLog, Log, TEXT("Process Ready!"));
        UE_LOG(GameServerLog, SetColor, TEXT("%s"), COLOR_NONE);
    }
    else
    {
        UE_LOG(GameServerLog, SetColor, TEXT("%s"), COLOR_RED);
        UE_LOG(GameServerLog, Log, TEXT("ERROR: Process Ready Failed!"));
        FGameLiftError ProcessReadyError = ProcessReadyOutcome.GetError();
        UE_LOG(GameServerLog, Log, TEXT("ERROR: %s"),
*ProcessReadyError.m_errorMessage);
        UE_LOG(GameServerLog, SetColor, TEXT("%s"), COLOR_NONE);
    }

    UE_LOG(GameServerLog, Log, TEXT("InitGameLift completed!"));
#endif
}
```

Fase 3: Ricostruisci il progetto del gioco

- Crea un progetto di gioco per entrambi i seguenti tipi di target: Development Editor e Development Server.

Note

Non è necessario ricostruire la soluzione. Invece, crea il progetto /Games/ nella cartella della tua app. In caso contrario, Visual Studio ricostruirà l'intero UE5 progetto, operazione che può richiedere fino a un'ora.

Package del tuo server di gioco per l'hosting

Ora che il codice del tuo server di gioco è integrato con la funzionalità SDK del server minima richiesta, sei pronto per creare un pacchetto di build del tuo server di gioco utilizzando Unreal Editor.

Per impacchettare la build del server di gioco

1. Apri il progetto di gioco nell'Unreal Editor.
2. Segui i passaggi dell'Unreal Editor per impacchettare il tuo server di gioco:
 - Scegli la tua piattaforma di destinazione (Windows o Linux).

- Seleziona l'obiettivo di build del server (*[your application name]*Server).

Il processo di impacchettamento genera il file eseguibile del server di gioco: *[your application name]*Server.exe.

3. Prepara la build del tuo server di gioco per la distribuzione su risorse di hosting. La build dovrebbe includere i seguenti file:

- Il tuo file eseguibile sul server di gioco
- Se utilizzi la versione 5.5 o precedente di Unreal Engine, includi i seguenti file per le build di Windows. Puoi trovarli nella versione originale di Unreal Engine:
 - VC_redist.x64.exe (UnrealEngine\Engine\Source\Programs\PrereqInstaller\Resources\VCRedist\)
 - UEPrereqSetup_x64.exe or UE5PrereqSetup_x64.exe (UnrealEngine\Engine\Extras\Redist\en-us\)
- Tutte le altre dipendenze richieste per il tuo server di gioco.
- Librerie OpenSSL, se necessario. Puoi saltare questo passaggio se il tuo server di gioco è integrato con la versione 5.3 o successiva dell'SDK Amazon GameLift Servers del server.
[L'ultima versione SDK del server è disponibile qui.](#)

Le librerie OpenSSL sono necessarie per i server di gioco creati con server SDK per Unreal, versioni 5.2 e precedenti

Devi includere la stessa versione delle librerie OpenSSL utilizzata per impacchettare il server di gioco in Unreal. Queste librerie si trovano nella fonte del motore di gioco. La posizione varia a seconda dell'ambiente di sviluppo:

In Windows:

- [ENGINE_ROOT_DIR]\Engine\Extras\ThirdPartyNotUE\libmobiledevice\x64\libssl-1_1-x64.dll
- [ENGINE_ROOT_DIR]\Engine\Extras\ThirdPartyNotUE\libmobiledevice\x64\libcrypto-1_1-x64.dll

In Linux:

- Engine/Source/Thirdparty/OpenSSL/1.1.1n/include/libssl.so.1.1

- Engine/Source/Thirdparty/openssl/1.1.1n/include/libcrypto.so.1.1

Copia le librerie OpenSSL nel pacchetto di build del gioco nella stessa directory del file eseguibile del server di gioco.

Fasi successive

Ora che hai preparato una build di server di gioco con le funzionalità minime richieste per l'hosting Amazon GameLift Servers, considera questi potenziali passaggi successivi:

- Implementa il tuo server di gioco integrato per il test e lo sviluppo. Con una flotta Anywhere, puoi configurare il tuo computer locale come risorsa di hosting e utilizzarlo per testare le connessioni del server di gioco e dei client di gioco. Per l'hosting basato sul cloud, distribuisce il tuo server di gioco su un EC2 gestito o su una flotta di container gestiti. Consulta questi argomenti come guida:
 - [Configurazione per lo sviluppo iterativo con Anywhere Amazon GameLift Servers](#)
 - [Amazon GameLift Servers Flotte ovunque](#)
 - [Amazon GameLift Servers EC2 flotte gestite](#)
 - [Amazon GameLift Servers flotte di container gestite](#)
- Personalizza l'integrazione del server di gioco aggiungendo funzionalità opzionali. Ad esempio, potresti voler aggiungere sessioni con giocatori unici IDs, impostare il matchmaking backfill o gestire l'accesso del server di gioco alle altre tue AWS risorse (come un database o un servizio di archiviazione dei contenuti). Consulta questi argomenti come guida:
 - [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#)
 - [Server C++ \(Unreal\) SDK 5.x per -- Azioni Amazon GameLift Servers](#)
- Personalizza il componente del client di gioco per richiedere sessioni di gioco, ricevere informazioni di connessione e connetterti direttamente a un server di gioco per giocare. Consulta questi argomenti come guida:
 - [Integra la funzionalità del client di Amazon GameLift Servers gioco](#)

Integrazione Amazon GameLift Servers in un progetto Unity

Scopri come integrare l'Amazon GameLift Servers SDK for Unity nei tuoi progetti di gioco per accedere al set completo di funzionalità SDK del server.

Suggerimento

Per una distribuzione più rapida, prova il plug-in Amazon GameLift Servers standalone per Unity. Fornisce flussi di lavoro guidati per l'interfaccia utente per implementare rapidamente il server di gioco con una configurazione minima, in modo da poter provare i componenti del gioco in azione. Per informazioni, consulta [Amazon GameLift Serversplugin per Unity \(server SDK 5.x\)](#).

Risorse aggiuntive:

- [Server C# SDK 5.x per -- Azioni Amazon GameLift Servers](#)
- [the section called “Ottieni strumenti di sviluppo”](#)

Installa il server SDK for Unity

Scarica l'open source Amazon GameLift Servers per Unity da [GitHub](#). I file readme del repository contengono i prerequisiti e le istruzioni di installazione.

Configura una flotta Amazon GameLift Servers Anywhere per i test

Puoi configurare la tua workstation di sviluppo come flotta di hosting Amazon GameLift Servers Anywhere per testare in modo iterativo la tua Amazon GameLift Servers integrazione. Con questa configurazione, puoi avviare i processi del server di gioco sulla tua workstation, inviare richieste di iscrizione ai giocatori o di matchmaking per Amazon GameLift Servers avviare sessioni di gioco e connettere i client alle nuove sessioni di gioco. Con la tua workstation configurata come server di hosting, puoi monitorare tutti gli aspetti dell'integrazione del gioco con Amazon GameLift Servers.

Per istruzioni sulla configurazione della workstation, consulta [Configura i test locali con Amazon GameLift Servers Anywhere](#) i seguenti passaggi:

1. Crea una posizione personalizzata per la tua workstation.
2. Crea una flotta Amazon GameLift Servers Anywhere con la tua nuova posizione personalizzata. In caso di successo, questa richiesta restituisce un ID della flotta. Prendi nota di questo valore, poiché ti servirà in seguito.
3. Registra la tua workstation come computer nella nuova flotta Anywhere. Fornisci un nome di elaborazione univoco e specifica l'indirizzo IP della tua workstation. In caso di successo, questa

richiesta restituisce un endpoint SDK del servizio, sotto forma di URL. WebSocket Prendi nota di questo valore, poiché ti servirà in seguito.

4. Genera un token di autenticazione per il calcolo della tua workstation. Questa autenticazione di breve durata include il token e una data di scadenza. Il server di gioco lo utilizza per autenticare la comunicazione con il Amazon GameLift Servers servizio. Archivia l'autenticazione sul computer della tua workstation in modo che i processi del server di gioco in esecuzione possano accedervi.

Aggiungi il codice Amazon GameLift Servers del server al tuo progetto Unity

Il tuo server di gioco comunica con il Amazon GameLift Servers servizio per ricevere istruzioni e segnalare lo stato in corso. A tale scopo, aggiungi il codice del server di gioco che utilizza l'SDK del Amazon GameLift Servers server.

L'esempio di codice fornito illustra gli elementi di integrazione di base richiesti. Usa a `MonoBehavior` per illustrare una semplice inizializzazione del server di gioco con. Amazon GameLift Servers L'esempio presuppone che il server di gioco funzioni su una flotta Amazon GameLift Servers Anywhere per i test. Include codice per:

- Inizializza un client Amazon GameLift Servers API. L'esempio utilizza la versione `InitSDK()` con parametri del server per la flotta e l'elaborazione di Anywhere. Utilizza l' WebSocket URL, l'ID della flotta, il nome di calcolo (ID host) e il token di autenticazione, come definito nell'argomento precedente. [Configura una flotta Amazon GameLift Servers Anywhere per i test](#)
- Implementa le funzioni di callback per rispondere alle richieste del Amazon GameLift Servers servizio, tra cui `OnStartGameSessionOnProcessTerminate`, e. `onHealthCheck`
- Chiama `ProcessReady()` con una porta designata per avvisare il Amazon GameLift Servers servizio quando il processo è pronto per ospitare sessioni di gioco.

Il codice di esempio fornito stabilisce la comunicazione con il Amazon GameLift Servers servizio. Implementa inoltre una serie di funzioni di callback che rispondono alle richieste del servizio. Amazon GameLift Servers Per ulteriori informazioni su ciascuna funzione e sul funzionamento del codice, consulta [Inizializzare il processo del server](#). Per ulteriori informazioni sulle azioni SDK e sui tipi di dati utilizzati in questo codice, leggi. [Server C# SDK 5.x per -- Azioni Amazon GameLift Servers](#)

Il codice di esempio mostra come aggiungere la funzionalità richiesta, come descritto in [Aggiungi Amazon GameLift Servers al tuo server di gioco](#). Per ulteriori informazioni sulle azioni dell'SDK del server, consulta [Server C# SDK 5.x per -- Azioni Amazon GameLift Servers](#).

Codice di esempio di integrazione

```
using System.Collections.Generic;
using Aws.GameLift.Server;
using UnityEngine;

public class ServerSDKManualTest : MonoBehaviour
{
    //This example is a simple integration that initializes a game server process
    //that is running on an Amazon GameLift Servers Anywhere fleet.
    void Start()
    {
        //Identify port number (hard coded here for simplicity) the game server is
        //listening on for player connections
        var listeningPort = 7777;

        //WebSocketUrl from RegisterHost call
        var websocketUrl = "wss://us-west-2.api.amazongamelift.com";

        //Unique identifier for this process
        var processId = "myProcess";

        //Unique identifier for your host that this process belongs to
        var hostId = "myHost";

        //Unique identifier for your fleet that this host belongs to
        var fleetId = "myFleet";

        //Authorization token for this host process
        var authToken = "myAuthToken";

        //Server parameters are required for an Amazon GameLift Servers Anywhere fleet.
        //They are not required for an Amazon GameLift Servers managed EC2 fleet.
        ServerParameters serverParameters = new ServerParameters(
            websocketUrl,
            processId,
            hostId,
            fleetId,
            authToken);

        //InitSDK establishes a local connection with an Amazon GameLift Servers agent
        //to enable further communication.
        var initSDKOutcome = GameLiftServerAPI.InitSDK(serverParameters);
        if (initSDKOutcome.Success)
```

```

{
    //Implement callback functions
    ProcessParameters processParameters = new ProcessParameters(
    //Implement OnStartGameSession callback
        (gameSession) => {
            //Amazon GameLift Servers sends a game session activation request
            to the game server
            //with game session object containing game properties and other
            settings.
            //Here is where a game server takes action based on the game
            session object.
            //When the game server is ready to receive incoming player
            connections,
            //it invokes the server SDK call ActivateGameSession().
            GameLiftServerAPI.ActivateGameSession();
        },
        (updateGameSession) => {
            //Amazon GameLift Servers sends a request when a game session is
            updated (such as for
            //FlexMatch backfill) with an updated game session object.
            //The game server can examine matchmakerData and handle new
            incoming players.
            //updateReason explains the purpose of the update.
        },
        () => {
            //Implement callback function OnProcessTerminate
            //Amazon GameLift Servers invokes this callback before shutting
            down the instance hosting this game server.
            //It gives the game server a chance to save its state, communicate
            with services, etc.,
            //and initiate shut down. When the game server is ready to shut
            down, it invokes the
            //server SDK call ProcessEnding() to tell Amazon GameLift Servers
            it is shutting down.
            GameLiftServerAPI.ProcessEnding();
        },
        () => {
            //Implement callback function OnHealthCheck
            //Amazon GameLift Servers invokes this callback approximately every
            60 seconds.
            //A game server might want to check the health of dependencies,
            etc.
            //Then it returns health status true if healthy, false otherwise.

```

```

        //The game server must respond within 60 seconds, or Amazon
        GameLift Servers records 'false'.
        //In this example, the game server always reports healthy.
        return true;
    },
    //The game server gets ready to report that it is ready to host game
sessions
    //and that it will listen on port 7777 for incoming player connections.
    listeningPort,
    new LogParameters(new List<string>()
    {
        //Here, the game server tells Amazon GameLift Servers where to find
game session log files.
        //At the end of a game session, Amazon GameLift Servers uploads
everything in the specified
        //location and stores it in the cloud for access later.
        "/local/game/logs/myserver.log"
    }));

    //The game server calls ProcessReady() to tell Amazon GameLift Servers it's
ready to host game sessions.
    var processReadyOutcome =
GameLiftServerAPI.ProcessReady(processParameters);
    if (processReadyOutcome.Success)
    {
        print("ProcessReady success.");
    }
    else
    {
        print("ProcessReady failure : " +
processReadyOutcome.Error.ToString());
    }
}
else
{
    print("InitSDK failure : " + initSDKOutcome.Error.ToString());
}
}

void OnApplicationQuit()
{
    //Make sure to call GameLiftServerAPI.ProcessEnding() and
GameLiftServerAPI.Destroy() before terminating the server process.

```

```
//These actions notify Amazon GameLift Servers that the process is terminating
and frees the API client from memory.
GenericOutcome processEndingOutcome = GameLiftServerAPI.ProcessEnding();
GameLiftServerAPI.Destroy();
if (processEndingOutcome.Success)
{
    Environment.Exit(0);
}
else
{
    Console.WriteLine("ProcessEnding() failed. Error: " +
processEndingOutcome.Error.ToString());
    Environment.Exit(-1);
}
}
```

Fasi successive

Ora che hai preparato una build di server di gioco con le funzionalità minime richieste per l'hosting Amazon GameLift Servers, considera questi potenziali passaggi successivi:

- Implementa il tuo server di gioco integrato per il test e lo sviluppo. Con una flotta Anywhere, puoi configurare il tuo computer locale come risorsa di hosting e utilizzarlo per testare le connessioni del server di gioco e dei client di gioco. Per l'hosting basato sul cloud, distribuisce il tuo server di gioco su un EC2 gestito o su una flotta di container gestiti. Consulta questi argomenti come guida:
 - [Configurazione per lo sviluppo iterativo con Anywhere Amazon GameLift Servers](#)
 - [Amazon GameLift Servers Flotte ovunque](#)
 - [Amazon GameLift Servers EC2 flotte gestite](#)
 - [Amazon GameLift Servers flotte di container gestite](#)
- Personalizza l'integrazione del server di gioco aggiungendo funzionalità opzionali. Ad esempio, potresti voler aggiungere sessioni con giocatori unici IDs, impostare il matchmaking backfill o gestire l'accesso del server di gioco alle altre tue AWS risorse (come un database o un servizio di archiviazione dei contenuti). Consulta questi argomenti come guida:
 - [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#)
 - [Server C++ \(Unreal\) SDK 5.x per -- Azioni Amazon GameLift Servers](#)

- Personalizza il componente del client di gioco per richiedere sessioni di gioco, ricevere informazioni di connessione e connetterti direttamente a un server di gioco per giocare. Consulta questi argomenti come guida:
 - [Integra la funzionalità del client di Amazon GameLift Servers gioco](#)

Package di un server di gioco costruito per la distribuzione

Prepara il software del tuo server di gioco per l'implementazione sulle tue risorse di hosting. Una volta distribuito, il software viene installato su ogni risorsa di hosting, quindi vengono avviati uno o più processi del server di gioco pronti per ospitare sessioni di gioco per i giocatori.

Preparare la build di un server di gioco varia a seconda del tipo di opzioni di Amazon GameLift Servers hosting utilizzate. Tutte le build dei server di gioco devono essere integrate con l'SDK del server per Amazon GameLift Servers, come descritto in [Integra un server di gioco con Amazon GameLift Servers](#)

Gli argomenti di questa sezione forniscono indicazioni su come preparare il software per la distribuzione nei seguenti scenari.

- Per l' EC2 hosting gestito, impacchettate il software del server e caricatelo Amazon GameLift Servers per la distribuzione.
- Per l'hosting gestito di container, crea un'immagine del contenitore con il software del server e archiviala in Amazon Elastic Container Registry per la distribuzione.
- Per l'hosting con Amazon GameLift Servers Anywhere, impacchetta il software del server secondo necessità per l'installazione sulle tue risorse di hosting.

Note

Se stai distribuendo uno script Amazon GameLift Servers Realtime configurato, consulta [Caricare uno script per Amazon GameLift ServersRealtime](#).

Argomenti

- [Crea un server di gioco costruito per Amazon GameLift Servers](#)
- [Crea un'immagine del contenitore per Amazon GameLift Servers](#)

Crea un server di gioco costruito per Amazon GameLift Servers

Dopo aver integrato il server di gioco con Amazon GameLift Servers (vedi [Prepara un gioco da ospitare con Amazon GameLift Servers](#)), installa il software del server di gioco sulle risorse di calcolo per l'hosting. Questo processo varia a seconda del tipo di Amazon GameLift Servers hosting che utilizzi.

Implementa per l'hosting gestito

Se utilizzi un hosting EC2 Amazon GameLift Servers gestito, devi impacchettare il software del server di gioco e caricarlo su. Amazon GameLift Servers Quando crei una flotta gestita, la distribuisce Amazon GameLift Servers automaticamente su ogni istanza del parco macchine.

Gli argomenti di questa sezione descrivono come impacchettare i file di build per il caricamento, creare uno script di installazione della build opzionale e quindi caricare i file utilizzando [AWS Command Line Interface \(AWS CLI\)](#) o AWS SDK.

Implementa l'hosting for Anywhere

Se utilizzi le flotte Amazon GameLift Servers Anywhere per l'hosting autogestito, è tua responsabilità installare il software del server di gioco su ogni computer del parco computer e mantenerlo aggiornato.

Quando un processo integrato del server di gioco inizia a funzionare, si inizializza automaticamente e stabilisce la comunicazione con il servizio. Amazon GameLift Servers Il processo del server avvia le sessioni di gioco in base alle richieste Amazon GameLift Servers e riporta l'attività al servizio.

Argomenti

- [Creare un pacchetto dei file della build di gioco](#)
- [Aggiungere uno script di installazione della build](#)
- [Crea una risorsa di Amazon GameLift Servers compilazione per l'hosting gestito](#)

Creare un pacchetto dei file della build di gioco

Prima di caricare il server di gioco configurato su Amazon GameLift Servers, impacchetta i file di build del gioco in una directory di compilazione. Questo processo è un requisito quando si ospita con una flotta gestita da EC2 ed è una buona pratica quando si ospita con una flotta Anywhere. La directory di build dovrebbe includere tutti i componenti necessari per far funzionare i server di gioco e ospitare sessioni di gioco. Ciò potrebbe includere quanto segue:

- **File binari del server di gioco:** i file binari necessari per eseguire il server di gioco. Una build può includere file binari per più server di gioco creati per funzionare sulla stessa piattaforma. Per un elenco delle piattaforme supportate, consulta [Ottieni strumenti Amazon GameLift Servers di sviluppo](#).
- **Dipendenze:** tutti i file dipendenti necessari per l'esecuzione degli eseguibili del server di gioco. Esempi di file dipendenti includono gli asset, i file di configurazione e le librerie dipendenti.

Note

Per le build di gioco create con il server SDK Amazon GameLift Servers per C++ (incluse quelle create con il plug-in Unreal), includi la DLL OpenSSL per la stessa versione di OpenSSL con cui hai creato l'SDK del server. Per maggiori dettagli, consulta il file README dell'SDK del server.

- **Script di installazione (opzionale):** un file di script per gestire le attività che installano la build del gioco sui Amazon GameLift Servers server di hosting. Posiziona questo file nella radice della directory di build. Amazon GameLift Servers esegue lo script di installazione come parte della creazione della flotta.

Puoi configurare qualsiasi applicazione della tua build, incluso lo script di installazione, per accedere alle tue risorse in modo sicuro su altri AWS servizi. Per informazioni su come eseguire questa operazione, consulta [Connect il server di gioco Amazon GameLift Servers ospitato ad altre AWS risorse](#).

Dopo aver impacchettato i file di build, assicurati che il server di gioco possa funzionare su un'installazione pulita del sistema operativo di destinazione per verificare che tutte le dipendenze richieste siano incluse e che lo script di installazione sia accurato.

Aggiungere uno script di installazione della build

Crea uno script di installazione per il sistema operativo (OS) della build del tuo gioco:

- **Windows:** crea un file batch denominato `install.bat`.
- **Linux:** crea un file di script di shell denominato `install.sh`.

Quando crei uno script di installazione, considera quanto segue:

- Lo script non può accettare alcun input da parte dell'utente.

- Amazon GameLift Servers installa la build e ricrea le directory dei file nel pacchetto di build su un server di hosting nelle seguenti posizioni:
 - Flotte Windows: C:\game
 - Flotte Linux: /local/game
- Durante il processo di installazione per le flotte Linux, l'utente run-as ha accesso limitato alla struttura dei file dell'istanza. Questo utente dispone dei diritti completi sulla directory in cui sono installati i file di build. Se lo script di installazione esegue azioni che richiedono le autorizzazioni di amministratore, specifica l'accesso da amministratore utilizzando `sudo`. Per impostazione predefinita, l'utente run-as for Windows Fleets dispone delle autorizzazioni di amministratore. Gli errori di autorizzazione correlati allo script di installazione generano un messaggio di evento che indica un problema dello script.
- Su Linux, Amazon GameLift Servers supporta i comuni linguaggi di interpretazione della shell come `bash`. Aggiungi uno shebang (ad esempio `#!/bin/bash`) all'inizio dello script di installazione. Per verificare il supporto per i comandi shell preferiti, accedi in remoto a un'istanza Linux attiva e apri un prompt della shell. Per ulteriori informazioni, consulta [Connect a flotte di istanze](#).
- Lo script di installazione non può fare affidamento su una connessione peering VPC. Una connessione peering VPC è disponibile solo dopo l'Amazon GameLift Servers installazione della build on fleet instances.

Example Windows installa il file bash

Questo `install.bat` file di esempio installa i componenti di runtime di Visual C++ necessari per il server di gioco e scrive i risultati in un file di registro. Lo script include il file del componente nel pacchetto di build alla radice.

```
vcredist_x64.exe /install /quiet /norestart /log c:\game\vcredist_2013_x64.log
```

Example Script della shell di installazione di Linux

Questo `install.sh` file di esempio utilizza `bash` nello script di installazione e scrive i risultati in un file di registro.

```
#!/bin/bash
echo 'Hello World' > install.log
```

Questo `install.sh` file di esempio mostra come utilizzare l' CloudWatch agente Amazon per raccogliere metriche a livello di sistema e personalizzate e gestire la rotazione dei log. Poiché

Amazon GameLift Servers viene eseguito in un VPC di servizio, devi concedere Amazon GameLift Servers le autorizzazioni per assumere un ruolo AWS Identity and Access Management (IAM) per tuo conto. Amazon GameLift Servers Per consentire l'assunzione di un ruolo, crea un ruolo che includa la policy AWS CloudWatchAgentAdminPolicy gestita e utilizza quel ruolo quando crei una flotta.

```
sudo yum install -y amazon-cloudwatch-agent
sudo yum install -y https://dl.fedoraproject.org/pub/epel/epel-release-
latest-7.noarch.rpm
sudo yum install -y collectd
cat <<'EOF' > /tmp/config.json
{
  "agent": {
    "metrics_collection_interval": 60,
    "run_as_user": "root",
    "credentials": {
      "role_arn": "arn:aws:iam::account#:role/rolename"
    }
  },
  "logs": {
    "logs_collected": {
      "files": {
        "collect_list": [
          {
            "file_path": "/tmp/log",
            "log_group_name": "gllog",
            "log_stream_name": "{instance_id}"
          }
        ]
      }
    }
  },
  "metrics": {
    "namespace": "GL_Metric",
    "append_dimensions": {
      "ImageId": "${aws:ImageId}",
      "InstanceId": "${aws:InstanceId}",
      "InstanceType": "${aws:InstanceType}"
    },
    "metrics_collected": {
      // Configure metrics you want to collect.
      // For more information, see Manually create or edit the CloudWatch agent configuration file.
    }
  }
}
```

```
}  
}  
EOF  
sudo mv /tmp/config.json /opt/aws/amazon-cloudwatch-agent/bin/config.json  
sudo /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl -a fetch-config -  
m ec2 -s -c file:/opt/aws/amazon-cloudwatch-agent/bin/config.json  
sudo systemctl enable amazon-cloudwatch-agent.service
```

Crea una risorsa di Amazon GameLift Servers compilazione per l'hosting gestito

Per creare una build e caricare i file, sono disponibili un paio di opzioni:

- [Crea una build da una directory di file](#). Questa è l'opzione più semplice e più comunemente usata.
- [Crea una build con file in Amazon Simple Storage Service \(Amazon S3\)](#). Con questa opzione, puoi gestire le tue versioni di build in Amazon S3.

Con entrambi i metodi, Amazon GameLift Servers crea una nuova risorsa di compilazione con un ID di build univoco e altri metadati. La build inizia con lo stato Inizializzato. Dopo aver Amazon GameLift Servers acquisito i file del server di gioco, la build passa allo stato Pronto.

Quando la build è pronta, puoi distribuirla su una nuova Amazon GameLift Servers flotta. Per ulteriori informazioni, consulta [Crea una EC2 flotta Amazon GameLift Servers gestita](#). Quando Amazon GameLift Servers configura la nuova flotta, scarica i file di build in ogni istanza del parco macchine e installa i file di build.

Crea una build da una directory di file

Per creare una build di gioco memorizzata in qualsiasi posizione, inclusa una directory locale, usa il [upload-build](#) AWS CLI comando. Questo comando crea il record di una nuova build in Amazon GameLift Servers e carica i file da un percorso specificato.

Inviare una richiesta di caricamento. In una finestra della riga di comando, inserisci il upload-build comando e i parametri seguenti.

```
aws gamelift upload-build \  
  --name user-defined name of build \  
  --operating-system supported OS \  
  --server-sdk-version server SDK for Amazon GameLift Servers version \  
  --build-root build path \  
  --build-version user-defined build number \  
  --build-artifact-path build artifact path
```

```
--region region name
```

- **operating-system**— L'ambiente di runtime della build del server di gioco. È necessario specificare un valore del sistema operativo. Non puoi aggiornarlo in un secondo momento.
- **server-sdk-version**— La versione dell'SDK del Amazon GameLift Servers server con cui è integrato il tuo server di gioco. Se non fornisci un valore, Amazon GameLift Servers utilizza il valore 4.0.2 predefinito. Se specifichi una versione SDK del server non corretta, la compilazione del server di gioco potrebbe non riuscire durante la chiamata `InitSdk` per stabilire una connessione al Amazon GameLift Servers servizio.
- **build-root**— Il percorso della directory dei file di build.
- **name**— Un nome descrittivo per la nuova build.
- **build-version**— I dettagli della versione per i file di build.
- **region**— La AWS regione in cui vuoi creare la tua build. Crea la build nella regione in cui intendi schierare le flotte. Se stai distribuendo il gioco in più regioni, crea una build in ogni regione.

Note

Visualizza la tua regione predefinita attuale utilizzando il [aws configure get region](#). Per modificare la tua regione predefinita, usa il [aws configure set region *region name*](#) comando.

Esempi

```
aws gamelift upload-build \  
  --operating-system AMAZON_LINUX_2023 \  
  --server-sdk-version "5.0.0" \  
  --build-root "~/mygame" \  
  --name "My Game Nightly Build" \  
  --build-version "build 255" \  
  --region us-west-2
```

```
aws gamelift upload-build \  
  --operating-system WINDOWS_2022 \  
  --server-sdk-version "5.0.0" \  
  --build-root "C:\mygame" \  
  --name "My Game Nightly Build" \  
  --build-version "build 255" \  
  --region us-east-1
```

```
--region us-west-2
```

In risposta alla richiesta di caricamento, Amazon GameLift Servers indica lo stato di avanzamento del caricamento. In caso di caricamento riuscito, Amazon GameLift Servers restituisce il nuovo ID del record di build. Il tempo di caricamento dipende dalle dimensioni dei file di gioco e dalla velocità di connessione.

Crea una build con file in Amazon S3

Puoi archiviare i tuoi file di build in Amazon S3 e caricarli Amazon GameLift Servers da lì. Quando crei la build, specifichi la posizione del bucket S3 e Amazon GameLift Servers recupera i file di build direttamente da Amazon S3.

Per creare una risorsa di compilazione

1. Archivia i tuoi file di build in Amazon S3. Crea un file.zip contenente i file di build compressi e caricalo in un bucket S3 del tuo Account AWS. Prendi nota dell'etichetta del bucket e del nome del file, ti serviranno per creare una build. Amazon GameLift Servers
2. Concedere ad Amazon GameLift Servers l'accesso ai file della build. Crea un ruolo IAM seguendo le istruzioni in [Accedi a un file di build del gioco in Amazon S3](#). Dopo aver creato il ruolo, prendi nota dell'Amazon Resource Name (ARN) del nuovo ruolo, ti servirà per creare una build.
3. Crea una build. Usa la Amazon GameLift Servers console o il AWS CLI per creare un nuovo record di build. È necessario disporre dell'`PassRole` autorizzazione, come descritto in [Esempi di autorizzazioni IAM per Amazon GameLift Servers](#).

Console

1. Nella [Amazon GameLift Servers console](#), nel pannello di navigazione, scegli Hosting, Builds.
2. Nella pagina Builds, scegli Crea build.
3. Nella pagina Crea build, in Impostazioni build, procedi come segue:
 - a. Per Nome, inserite il nome dello script.
 - b. In Versione, inserisci una versione. Poiché puoi aggiornare il contenuto di una build, i dati sulla versione possono aiutarti a tenere traccia degli aggiornamenti.
 - c. Per Sistema operativo (OS), scegli il sistema operativo della build del tuo server di gioco. Non puoi aggiornare questo valore in un secondo momento.

- d. Per la build del server di gioco, inserisci l'URI S3 dell'oggetto di build che hai caricato su Amazon S3 e scegli la versione dell'oggetto. Se non ricordi l'URI e la versione dell'oggetto di Amazon S3, scegli Browse S3 e cerca l'oggetto di compilazione.
 - e. Per il ruolo IAM, scegli il ruolo che hai creato che dà Amazon GameLift Servers accesso al tuo bucket S3 e all'oggetto di compilazione.
4. (Facoltativo) In Tag, aggiungi i tag alla build inserendo le coppie Chiave e Valore.
 5. Scegli Create (Crea).

Amazon GameLift Servers assegna un ID alla nuova build e carica il file.zip designato. Puoi visualizzare la nuova build, incluso lo stato, nella pagina Builds.

AWS CLI

Per definire la nuova build e caricare i file di build del server, usa il [create-build](#) comando.

1. Apri una finestra della riga di comando e passa a una directory in cui puoi usare AWS CLI.
2. Immettete il seguente create-build comando:

```
aws gamelift create-build \
  --name user-defined name of build \
  --server-sdk-version server SDK for Amazon GameLift Servers version \
  --operating-system supported OS \
  --build-version user-defined build number \
  --storage-location "Bucket"=S3 bucket label, "Key"=Build .zip file name, "RoleArn"=Access role ARN} \
  --region region name
```

- name— Un nome descrittivo per la nuova build.
- server-sdk-version— La versione del server SDK Amazon GameLift Servers che hai usato con cui integrare il tuo server di gioco. Amazon GameLift Servers Se non fornisci un valore, Amazon GameLift Servers utilizza il valore 4.0.2 predefinito.
- operating-system— L'ambiente di runtime della build del server di gioco. È necessario specificare un valore del sistema operativo. Non puoi aggiornarlo in un secondo momento.
- build-version— I dettagli della versione per i file di build. Queste informazioni possono essere utili perché ogni nuova versione del server di gioco richiede una nuova risorsa di build.
- storage-location

- **Bucket**— Il nome del bucket S3 che contiene la build. Ad esempio, «my_build_files».
- **Key**— Il nome del file.zip che contiene i file di build. Ad esempio, «my_game_build_7.0.1, 7.0.2».
- **RoleARN**— L'ARN assegnato al ruolo IAM che hai creato. Ad esempio, «arn:aws:iam: :111122223333:role/». GameLiftAccess Per un esempio di policy, consulta [Accedi a un file di build del gioco in Amazon S3](#).
- **region**— Crea la build nella regione in cui intendi schierare le flotte. AWS Se stai distribuendo il gioco in più regioni, crea una build in ogni regione.

 Note

Ti consigliamo di controllare la tua regione predefinita attuale usando il [configure get](#) comando . Per cambiare la tua regione predefinita, usa il [configure set](#) comando.

Esempio

```
aws gamelift create-build \  
  --operating-system WINDOWS_2022 \  
  --storage-location  
  "Bucket"="my_game_build_files", "Key"="mygame_build_101.zip", "RoleArn"="arn:aws:iam::111122223333:role/gamelift" \  
  --name "My Game Nightly Build" \  
  --build-version "build 101" \  
  --region us-west-2
```

3. Per visualizzare la nuova build, usa il [describe-build](#) comando.

Crea un'immagine del contenitore per Amazon GameLift Servers

Questo argomento descrive come creare un'immagine del contenitore con il software del server di gioco con Amazon GameLift Servers cui utilizzarla. L'immagine del contenitore del server di gioco include l'eseguibile del server di gioco e tutte le dipendenze necessarie per l'esecuzione. Le immagini dei container del server di gioco vengono utilizzate con una soluzione di hosting di container Amazon GameLift Servers gestiti. Per i dettagli sulla creazione della soluzione completa, consulta:

- [Roadmap di sviluppo per l'hosting con contenitori Amazon GameLift Servers gestiti](#)

- [Come funzionano i contenitori Amazon GameLift Servers](#)

Completa le seguenti attività per preparare l'immagine del container del server di gioco per la distribuzione in una flotta di Amazon GameLift Servers container. Prima di iniziare queste attività, completa l'integrazione del codice del server di gioco con l'SDK del Amazon GameLift Servers server.

Argomenti

- [Crea un'immagine del contenitore del server di gioco](#)
- [Invia l'immagine di un contenitore ad Amazon ECR](#)

Crea un'immagine del contenitore del server di gioco

Segui queste istruzioni mentre lavori su una piattaforma basata su Linux o usi Windows Subsystem for Linux (WSL) con Docker installato.

Per creare un'immagine del contenitore del server di gioco

1. Prepara una directory di lavoro con il software del tuo server di gioco. Su un computer locale, crea una directory di lavoro per organizzare i file per il contenitore del tuo server di gioco. L'immagine del contenitore utilizza questa struttura di file quando distribuisce il contenitore su Amazon GameLift Servers risorse per l'hosting. Esempio:

```
[~/]$ mkdir -p work/glc/gamebuild && cd work && find .  
.  
./glc  
./glc/gamebuild
```

Note

Se stai provando questa funzionalità e non hai ancora un server di gioco funzionante, prova il nostro server di gioco di esempio [SimpleServer](#), disponibile su GitHub.

2. Crea un nuovo Dockerfile utilizzando il modello fornito.
3. Segui le istruzioni nel modello Dockerfile per aggiornarlo per uso personale.
 - Aggiorna l'immagine di base secondo necessità.
 - Imposta le variabili di ambiente per la build del tuo server di gioco.

4. Crea l'immagine del contenitore. Esegui `docker build`, specificando il nome del tuo repository. Esempio:

```
[~/work/glc]$ docker build -t <local repository name>:<optional tag> .
```

È possibile visualizzare i repository e le immagini IDs utilizzando il `docker images` comando, come illustrato in questo esempio:

Modello Dockerfile per l'immagine del contenitore di un server di gioco

Questo modello contiene le istruzioni minime di cui un contenitore di server di gioco ha bisogno per essere utilizzabile in una Amazon GameLift Servers flotta. Modifica il contenuto in base alle esigenze del tuo server di gioco.

```
# Base image
# -----
# Add the base image that you want to use,
# Make sure to use an image with the same architecture as the
# Instance type you are planning to use on your fleets.
FROM public.ecr.aws/amazonlinux/amazonlinux
#
# Game build directory
# -----
# Add your gamebuild directory to the env variable below.
# The game build provided here needs to be integrated with server sdk for Amazon
# GameLift Servers.
ENV GAME_BUILD_DIRECTORY="<ADD_GAME_BUILD_DIRECTORY>" \
#
# Game executable and launch parameters
# -----
# Add the relative path to your executable in the 'GAME_EXECUTABLE' env variable
# below.
# The game build provided over here needs to be integrated with server sdk for Amazon
# GameLift Servers.
# This template assumes that the executable path is relative to the game build
# directory.
# Add any launch parameters to pass into your executable in the 'LAUNCH_PARAMS' env
# variable below.
# Add 'HOME_DIR' to identify where the game executable and logs exist.
GAME_EXECUTABLE="<ADD NAME OF EXECUTABLE WITHIN THE GAME DIRECTORY>" \
LAUNCH_PARAMS=<ADD LAUNCH PARAMETERS> \
```



```
HOME_DIR="/local/game" \  
  
# Install dependencies as necessary  
RUN yum install -y shadow-utils  
  
RUN mkdir -p $HOME_DIR  
COPY ./$GAME_BUILD_DIRECTORY/ $HOME_DIR  
  
# Change directory to home  
WORKDIR $HOME_DIR  
  
# Set up for 'gamelift' user  
RUN useradd -m gamelift && \  
    echo "gamelift ALL=(ALL) NOPASSWD:ALL" >> /etc/sudoers && \  
    chown -R gamelift:gamelift $HOME_DIR  
  
# Add permissions to game build  
RUN chmod +x ./$GAME_EXECUTABLE  
  
USER gamelift  
  
# Check directory before starting the container  
RUN ls -lhrt .  
  
# Check path before starting the container  
RUN echo $PATH  
  
# Start the game build  
ENTRYPOINT ["/bin/sh", "-c", ".$GAME_EXECUTABLE", "$LAUNCH_PARAMS"]
```

Invia l'immagine di un contenitore ad Amazon ECR

Dopo aver creato un'immagine del contenitore per la distribuzione Amazon GameLift Servers, archivia l'immagine in un repository pubblico o privato in Amazon ECR. A questo repository viene assegnato un valore URI, che viene Amazon GameLift Servers utilizzato per scattare un'istantanea dell'immagine per la distribuzione in una flotta di container.

 Note

Se non disponi ancora di un repository privato Amazon ECR, [creane](#) uno.

Per inviare l'immagine del contenitore ad Amazon ECR

1. Ottieni le tue credenziali Amazon ECR. Prima di inviare un'immagine del contenitore ad Amazon ECR, acquisisci AWS le tue credenziali in formato temporaneo e forniscile a Docker. Docker necessita di queste credenziali per accedere.

```
[~/work/glc]$ aws ecr get-login-password --region us-west-2 | docker login --  
username AWS --password-stdin aws_account_id.dkr.ecr.us-west-2.amazonaws.com  
WARNING! Your password will be stored unencrypted in  
/home/user-name/.docker/config.json.  
Configure a credential helper to remove this warning.  
See https://docs.docker.com/engine/reference/commandline/login/#credentials-store  
  
Login Succeeded
```

2. Copia l'URI del [repository privato Amazon ECR](#) che desideri utilizzare.
3. Applica un tag Amazon ECR all'immagine del contenitore.

Example

```
[~/work/glc]$ docker tag <IMAGE ID from above> <Amazon ECR private repository  
URI>:<optional tag>
```

4. Invia l'immagine del contenitore ad Amazon ECR

Example

```
[~/work/glc]$ docker image push <Amazon ECR private repository URI>
```

client/server Interazioni di gioco con Amazon GameLift Servers

I componenti della soluzione di Amazon GameLift Servers hosting interagiscono tra loro in modi specifici per eseguire sessioni di gioco in risposta alla domanda dei giocatori. Questo argomento descrive come i componenti comunicano tra loro quando il server di gioco è ospitato su flotte EC2 Amazon GameLift Servers gestite, flotte Amazon GameLift Servers Anywhere autogestite o su una soluzione ibrida.

I componenti della soluzione di hosting includono un server di gioco, il Amazon GameLift Servers servizio, un servizio di backend lato client e un client di gioco. Il server di gioco utilizza l'[SDK del Amazon GameLift Servers server](#) per interagire con il servizio. Amazon GameLift Servers Il servizio di

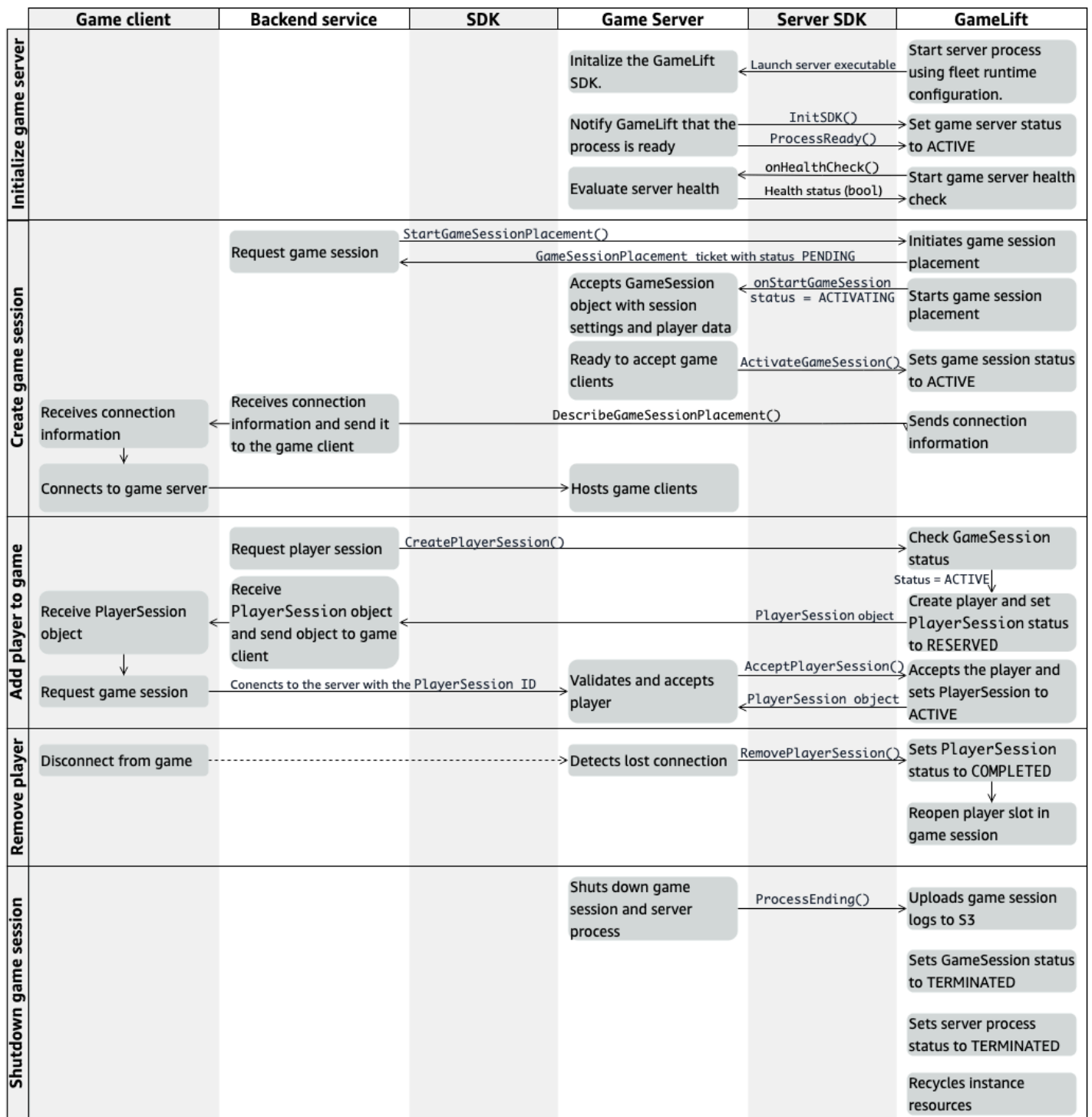
backend utilizza l'[API del Amazon GameLift Servers servizio](#) (parte dell' AWS SDK) per interagire con il servizio per conto del client di gioco. Quando si accede a una sessione di gioco, il client di gioco si connette direttamente a una sessione di gioco utilizzando l'indirizzo IP e il numero di porta univoci della sessione di gioco.

Argomenti

- [Diagramma delle interazioni](#)
- [Comportamenti di interazione](#)

Diagramma delle interazioni

Il diagramma seguente illustra come i componenti di hosting dei giochi interagiscono in modo che il Amazon GameLift Servers servizio possa tracciare lo stato della disponibilità del server di gioco e avviare sessioni di gioco in risposta alle richieste dei giocatori.



Comportamenti di interazione

Le sezioni seguenti descrivono la sequenza di eventi in ciascuna delle interazioni chiave.

Inizializzazione di un processo del server di gioco

All'avvio, un processo del server di gioco stabilisce la comunicazione con il Amazon GameLift Servers servizio e segnala che è pronto per ospitare una sessione di gioco.

1. Un nuovo processo eseguibile del server di gioco inizia a funzionare su una risorsa di hosting.
2. Il processo del server di gioco richiama le seguenti operazioni SDK del server in sequenza:
 - a. `InitSDK()` per inizializzare l'SDK del server, autenticare il processo del server e stabilire una comunicazione con il servizio. Amazon GameLift Servers
 - b. `ProcessReady()` per comunicare la disponibilità a ospitare una sessione di gioco. Questa chiamata riporta anche le informazioni di connessione del processo, quali client di gioco utilizzano per connettersi alla sessione di gioco e altre informazioni.

Il processo del server attende quindi le richieste del servizio. Amazon GameLift Servers

3. Amazon GameLift Servers aggiorna lo stato del processo del server ACTIVE e lo rende disponibile per ospitare una nuova sessione di gioco.
4. Amazon GameLift Servers inizia a richiamare periodicamente il `onHealthCheck` callback per richiedere lo stato di integrità dei processi del server. Queste chiamate continuano mentre il processo del server rimane attivo. Il processo del server deve rispondere correttamente o meno entro un minuto. Se il processo del server non risponde correttamente o non risponde, a un certo punto il Amazon GameLift Servers servizio cambia lo stato attivo del processo server e interrompe l'invio di richieste per l'avvio della sessione di gioco.

Creare una sessione di gioco

Il Amazon GameLift Servers servizio avvia una nuova sessione di gioco in risposta alla richiesta di un giocatore di giocare.

1. Un giocatore che utilizza il client di gioco chiede di partecipare a una sessione di gioco. A seconda di come il gioco gestisce la procedura di accesso dei giocatori, il client di gioco invia una richiesta al servizio di backend.
2. Se la procedura di accesso del giocatore richiede l'avvio di una nuova sessione di gioco, il servizio di backend invia una richiesta per una nuova sessione di gioco al Amazon GameLift Servers servizio. Questa richiesta richiama l'operazione `StartGameSessionPlacement()` dell'API del servizio. (In alternativa, il servizio di backend potrebbe chiamare `StartMatchmaking()` o `CreateGameSession()`.)

3. Il Amazon GameLift Servers servizio risponde creando un nuovo `GameSessionPlacement` ticket con lo stato `PENDING`. Restituisce le informazioni sul ticket al servizio di backend, in modo che possa tenere traccia dello stato del ticket di piazzamento e determinare quando la sessione di gioco è pronta per i giocatori. Per ulteriori informazioni, consulta [Imposta la notifica degli eventi per il posizionamento della sessione di gioco](#).
4. Il Amazon GameLift Servers servizio avvia il processo di posizionamento della sessione di gioco. Identifica le flotte da esaminare e cerca in esse un processo server attivo che non ospita una sessione di gioco. Quando individua un processo server disponibile, il servizio esegue le Amazon GameLift Servers seguenti operazioni:
 - a. Crea un `GameSession` oggetto con le impostazioni della sessione di gioco e i dati del giocatore contenuti nella richiesta di posizionamento e imposta lo stato su `ACTIVATING`.
 - b. Richiede al processo del server di avviare una sessione di gioco. Il servizio richiama il `onStartGameSession` callback del processo server e passa l'oggetto `GameSession`.
 - c. Modifica il numero di sessioni di gioco del processo del server in `1`.
5. Il processo server esegue la sua funzione di `onStartGameSession` callback. Quando il processo server è pronto per accettare le connessioni dei giocatori, richiama l'operazione SDK del server `ActivateGameSession()` e attende le connessioni del giocatore.
6. Il Amazon GameLift Servers servizio aggiorna l'`GameSession` oggetto con le informazioni di connessione per il processo del server (come riportato nella chiamata `aProcessReady()`) e imposta lo stato della sessione di gioco su `ACTIVE`. Inoltre aggiorna lo stato `GameSessionPlacement` del ticket a `FULFILLED`.
7. Il servizio di backend chiama `DescribeGameSessionPlacement()` per controllare lo stato del ticket e ottenere informazioni sulla sessione di gioco. Quando la sessione di gioco è attiva, il servizio di backend notifica il client di gioco e trasmette le informazioni di connessione alla sessione di gioco.
8. Il client di gioco utilizza le informazioni di connessione per connettersi direttamente al processo del server di gioco e partecipare alla sessione di gioco.

Aggiungere un giocatore a una partita

Un gioco può opzionalmente utilizzare le sessioni dei giocatori per tenere traccia delle connessioni dei giocatori alle sessioni di gioco. Le sessioni dei giocatori possono essere create individualmente o come parte di una richiesta di posizionamento per le sessioni di gioco.

1. Il servizio di backend richiama l'operazione dell'API del servizio `CreatePlayerSession()` con un ID di sessione di gioco.
2. Il Amazon GameLift Servers servizio controlla lo stato della sessione di gioco (deve essere così `ACTIVE`) e cerca uno slot per giocatori libero nella sessione di gioco. Se è disponibile uno slot, il servizio esegue le seguenti operazioni:
 - a. Crea un nuovo `PlayerSession` oggetto e imposta lo stato su `RESERVED`.
 - b. Risponde alla richiesta del servizio di backend con informazioni sulla sessione del giocatore.
3. Il servizio di backend trasmette le informazioni sulla sessione del giocatore al client di gioco insieme alle informazioni sulla connessione della sessione di gioco.
4. Il client di gioco utilizza le informazioni di connessione e l'ID della sessione del giocatore per connettersi direttamente al processo del server di gioco e chiedere di partecipare alla sessione di gioco.
5. In risposta a un tentativo di accesso al client di gioco, il processo del server di gioco richiama l'API del servizio `AcceptPlayerSession()` per convalidare l'ID di sessione del giocatore. Il processo server accetta o rifiuta la connessione.
6. Il Amazon GameLift Servers servizio esegue una delle seguenti operazioni:
 - a. Se la connessione viene accettata, Amazon GameLift Servers imposta lo `PlayerSession` stato su `ACTIVE` e lo passa `PlayerSession` al processo del server di gioco.
 - b. Se il processo del server di gioco non richiede `AcceptPlayerSession()` l'ID della sessione del giocatore entro un certo periodo di tempo dalla `CreatePlayerSession()` richiesta originale, il Amazon GameLift Servers servizio cambia `PlayerSession` lo stato `TIMEDOUT` e riapre lo slot del giocatore nella sessione di gioco.

Rimuovere un giocatore

Per i giochi che utilizzano sessioni di gioco, il processo del server di gioco notifica al Amazon GameLift Servers servizio quando un giocatore si disconnette. Il servizio utilizza queste informazioni per tenere traccia dello stato degli slot dei giocatori in una sessione di gioco e può consentire ai nuovi giocatori di utilizzare gli slot aperti.

1. Un giocatore si disconnette dalla sessione di gioco.
2. Il processo del server di gioco rileva la connessione persa e richiama l'operazione SDK del server. `RemovePlayerSession()`

3. Il Amazon GameLift Servers servizio modifica lo stato della sessione del giocatore COMPLETED e riapre lo slot del giocatore nella sessione di gioco.

Chiusura della sessione di gioco

Al termine di una sessione di gioco o quando si chiude la sessione di gioco, il processo del server notifica al Amazon GameLift Servers servizio lo stato della sessione di gioco.

1. Il processo del server di gioco termina la sessione di gioco e avvia l'arresto del processo richiamando l'operazione SDK del server. `ProcessEnding()`
2. Il server di gioco elabora le chiamate `Destroy()` per liberare l'SDK del server dalla memoria. Questo passaggio è necessario per evitare che le [metriche di telemetria](#) segnalino le normali uscite dal processo come arresti anomali.
3. Il servizio esegue le seguenti operazioni: Amazon GameLift Servers
 - a. Carica i log delle sessioni di gioco su Amazon Simple Storage Service (Amazon S3).
 - b. Modifica lo stato della sessione di gioco in. `TERMINATED`
 - c. Modifica lo stato del processo del server in `TERMINATED`.
 - d. A seconda di come è progettata la soluzione di hosting, le nuove risorse di hosting disponibili vengono allocate per eseguire un nuovo processo del server di gioco.

Crea un servizio di backend per Amazon GameLift Servers

Ti consigliamo di implementare un servizio client di gioco che autentichi i tuoi giocatori e comunichi con l'API. Amazon GameLift Servers Implementando un servizio client di gioco personalizzato, puoi:

- Personalizza l'autenticazione per i tuoi giocatori.
- Controlla il modo in cui Amazon GameLift Servers raggruppa i giocatori per nuove sessioni di gioco o per aggiungerle a sessioni di gioco esistenti.
- Raccogli informazioni dalle tue risorse per fornire attributi di gioco ai giocatori come la valutazione delle abilità per il matchmaking invece di fidarti del cliente.

L'utilizzo di un servizio client di gioco riduce anche i rischi per la sicurezza introdotti dai client di gioco che interagiscono direttamente con la tua API. Amazon GameLift Servers

Autenticazione dei giocatori

Puoi utilizzare Amazon Cognito e la sessione giocatore IDs per autenticare i tuoi client di gioco. Per gestire il ciclo di vita e le proprietà delle identità dei giocatori, utilizza i pool di utenti di Amazon Cognito.

Se preferisci, crea una soluzione di identità personalizzata e ospitala su AWS. Puoi anche utilizzare gli autorizzatori Lambda per la logica di autorizzazione personalizzata con API Gateway.

Risorse aggiuntive:

- [Utilizzo di pool di identità \(identità federate\)](#) (Amazon Cognito Developer Guide)
- [Guida introduttiva ai pool di utenti](#) (Amazon Cognito Developer Guide)
- [Come configurare l'autenticazione dei giocatori con Amazon Cognito](#) (AWS per il blog dei giochi)

Integra la funzionalità del client di Amazon GameLift Servers gioco

Integra la funzionalità di hosting dei Amazon GameLift Servers giochi in tutti i componenti della soluzione che devono comunicare con il Amazon GameLift Servers servizio, ad esempio per ottenere informazioni sulle sessioni di gioco o creare nuove sessioni di gioco. Nella maggior parte dei casi, questa funzionalità è integrata nel componente del servizio di backend. Aggiungi funzionalità per le attività principali, tra cui:

- Richiedi informazioni e lo stato delle sessioni di gioco attive.
- Riserva uno slot per un nuovo giocatore in una sessione di gioco esistente.
- Crea una nuova sessione di gioco per un gruppo di giocatori.
- Richiesta di abbinamento per uno o più giocatori.
- Fornisci dati aggiornati per le sessioni di gioco esistenti.

Vedi [client/server Interazioni di gioco con Amazon GameLift Servers](#) per ulteriori dettagli su come i componenti di hosting dei giochi interagiscono tramite Amazon GameLift Servers SDKs.

Configura l'Amazon GameLift ServersAPI

Aggiungi il AWS SDK per C++ con Amazon GameLift Servers a un progetto.

Aggiungi codice per inizializzare un Amazon GameLift Servers client e memorizzare le impostazioni chiave. Questo codice deve essere eseguito prima di qualsiasi codice dipendente Amazon GameLift Servers da.

1. Imposta una configurazione client. Utilizza la configurazione client predefinita o crea un oggetto di configurazione client personalizzato. Per ulteriori informazioni, vedere [AWS::Client::ClientConfiguration](#)(C++) o [AmazonGameLiftConfig](#)(C#).

Una configurazione client specifica una regione e un endpoint di destinazione da utilizzare per i contatti. Amazon GameLift Servers La regione identifica l'insieme di risorse distribuite (flotte, code e matchmaker) da utilizzare. La configurazione client predefinita imposta la posizione nella regione Stati Uniti orientali (Virginia settentrionale). Per utilizzare qualsiasi altra regione, crea una configurazione personalizzata.

2. Inizializzare un client Amazon GameLift Servers. Usa [Aws:GameLift:: GameLiftClient \(\)](#) (C++) o [AmazonGameLiftClient\(\)](#) (C#) con una configurazione client predefinita o una configurazione client personalizzata.
3. Se utilizzi sessioni di gioco, aggiungi un meccanismo per generare un identificatore univoco per ogni giocatore. Per ulteriori informazioni, consulta [Genera giocatore IDs](#).
4. Raccogli e archivia le seguenti informazioni:
 - Flotta bersaglio: se stai creando manualmente sessioni di gioco su una flotta specifica, fornisci un ID flotta o un ID alias che punti alla flotta bersaglio. Se la flotta è composta da più sedi, specifica un'ubicazione della flotta. Come best practice, utilizza gli alias della flotta in modo da poter passare i giocatori da una flotta all'altra senza dover aggiornare il servizio di backend.
 - Coda di destinazione: per i giochi che utilizzano code con più flotte per effettuare nuove sessioni di gioco, specifica il nome della coda a cui inviare le richieste di posizionamento.
 - AWS credenziali: tutte le chiamate Amazon GameLift Servers devono fornire le credenziali dell'host del gioco. Account AWS È possibile acquisire queste credenziali creando un utente giocatore, come descritto in. [Configura l'accesso programmatico per il tuo gioco](#) A seconda di come gestisci l'accesso per l'utente giocatore, procedi come segue:
 - Se utilizzi un ruolo per gestire le autorizzazioni utente del giocatore, aggiungi il codice per assumere il ruolo prima di chiamare un'Amazon GameLift ServersAPI. La richiesta di assunzione del ruolo restituisce una serie di credenziali di sicurezza temporanee. Per ulteriori informazioni, consulta [Switching to an IAM role \(AWS API\)](#) nella IAM User Guide.
 - Se disponi di credenziali di sicurezza a lungo termine, configura il codice per individuare e utilizzare le credenziali archiviate. Vedi [Autenticazione tramite credenziali a lungo termine](#)

[nella Guida](#) di riferimento agli strumenti AWS SDKs e strumenti. [Per informazioni sulla memorizzazione delle credenziali, consulta i riferimenti AWS API per \(C++\) e \(.NET\).](#)

- Se disponi di credenziali di sicurezza temporanee, aggiungi il codice per aggiornare regolarmente le credenziali utilizzando AWS Security Token Service (AWS STS), come descritto in [Utilizzo delle credenziali di sicurezza temporanee con la Guida per l' AWS SDKs](#) utente IAM. Il codice deve richiedere nuove credenziali prima della scadenza di quelle precedenti.

Ottieni sessioni di gioco attive

Aggiungi codice per scoprire le sessioni di gioco disponibili e gestire le impostazioni e i metadati delle sessioni di gioco.

Cerca sessioni di gioco attive

Consente [SearchGameSessions](#) di ottenere informazioni su una sessione di gioco specifica, su tutte le sessioni attive o sulle sessioni che soddisfano una serie di criteri di ricerca. Questa chiamata restituisce un [GameSession](#) oggetto per ogni sessione di gioco attiva che corrisponde alla richiesta di ricerca. Questo oggetto contiene il nome DNS, l'indirizzo IP e la porta di cui un client di gioco ha bisogno per connettersi alla sessione di gioco.

Utilizza i criteri di ricerca per ottenere un elenco filtrato di sessioni di gioco attive a cui possono unirsi i giocatori. Ad esempio, è possibile filtrare le sessioni come segue:

- Escludi le sessioni di gioco complete: `CurrentPlayerSessionCount = MaximumPlayerSessionCount`.
- Scegli le sessioni di gioco in base alla durata della sessione: `ValutaCreationTime`.
- Trova sessioni di gioco in base a una proprietà di gioco personalizzata: `gameSessionProperties.gameMode = "brawl"`.

Gestisci i dati delle sessioni di gioco

Utilizzare una delle operazioni elencate di seguito per recuperare o aggiornare le informazioni sulla sessione di gioco.

- [DescribeGameSessionDetails\(\)](#) — Visualizza lo stato di protezione di una sessione di gioco oltre alle informazioni sulla sessione di gioco.

- [UpdateGameSession\(\)](#) — Modifica i metadati e le impostazioni di una sessione di gioco in base alle esigenze.
- [GetGameSessionLogUrl](#) — Accedi ai registri delle sessioni di gioco memorizzati.

Crea sessioni di gioco

Aggiungere codice per avviare nuove sessioni di gioco nei parchi di istanze distribuite e renderle disponibili ai giocatori. Esistono due opzioni per creare sessioni di gioco, a seconda di come la soluzione di hosting di giochi gestisce i posizionamenti delle sessioni di gioco.

Crea una sessione di gioco con una coda in più sedi

Si usa [StartGameSessionPlacement](#) per inserire una richiesta per una nuova sessione di gioco in una coda. Per utilizzare questa operazione, crea una coda. Questo determina dove Amazon GameLift Servers colloca la nuova sessione di gioco. Per ulteriori informazioni sulle code e su come utilizzarle, consulta [Configura il posizionamento delle sessioni di gioco](#).

Quando crei un posizionamento per una sessione di gioco, specifica il nome della coda da utilizzare, un nome per la sessione di gioco, un numero massimo di giocatori simultanei e un set opzionale di proprietà del gioco. Facoltativamente, puoi anche fornire un elenco di giocatori a cui partecipare automaticamente alla sessione di gioco. Se includi i dati sulla latenza dei giocatori per le regioni pertinenti, Amazon GameLift Servers utilizza queste informazioni per collocare la nuova sessione di gioco su una flotta che offra l'esperienza di gioco ideale per i giocatori.

Per ottenere misurazioni accurate della latenza, utilizza i beacon Amazon GameLift Servers di ping UDP. Questi endpoint consentono di misurare l'effettiva latenza della rete UDP tra i dispositivi dei giocatori e le potenziali sedi di hosting, con il risultato di decisioni di posizionamento più accurate rispetto all'utilizzo dei ping ICMP. Per ulteriori informazioni sull'utilizzo dei beacon ping UDP per misurare la latenza, fare riferimento a [UDP ping beacon](#).

Il posizionamento delle sessioni di gioco è un processo asincrono. Dopo aver effettuato una richiesta, puoi lasciarla andare a buon fine o scadere. Puoi anche annullare la richiesta in qualsiasi momento utilizzando [StopGameSessionPlacement](#). Per verificare lo stato della tua richiesta di collocamento, chiama [DescribeGameSessionPlacement](#).

Crea una sessione di gioco su una flotta specifica

Usa [CreateGameSession](#) per creare una nuova sessione su una flotta specificata. Questa operazione sincrona riesce o non riesce a seconda che il parco disponga di risorse disponibili per ospitare una

nuova sessione di gioco. Dopo aver Amazon GameLift Servers creato la nuova sessione di gioco e aver [GameSession](#) restituito un oggetto, puoi unirvi altri giocatori.

Quando usi questa operazione, fornisci un ID flotta o un ID alias, un nome di sessione e il numero massimo di giocatori simultanei per il gioco. In alternativa, è possibile includere un set di proprietà di gioco. Le proprietà del gioco sono definite in una serie di coppie chiave-valore.

Se utilizzi la funzionalità di protezione Amazon GameLift Servers delle risorse per limitare il numero di sessioni di gioco che un giocatore può creare, fornisci l'ID giocatore del creatore della sessione di gioco.

Unisciti a un giocatore a una sessione di gioco

Aggiungi codice per riservare uno slot per giocatore in una sessione di gioco attiva e connettere i client di gioco alle sessioni di gioco. Questa azione è disponibile se il gioco utilizza sessioni di gioco con un solo giocatore IDs. Per ulteriori informazioni sulle sessioni dei giocatori, consulta [Amazon GameLift Servere l'esperienza del giocatore](#).

1. Riserva uno slot per giocatore in una sessione di gioco

Per riservare uno slot, crea una sessione giocatore per la sessione di gioco. Ci sono due modi per effettuare questa operazione:

- Se utilizzi [StartGameSessionPlacement](#) vuoi [StartMatchmaking](#) creare una sessione di gioco, puoi includere uno o più giocatori nella richiesta di creare sessioni di gioco per loro nella nuova sessione di gioco.
- Per aggiungere giocatori a una sessione di gioco esistente, chiama [CreatePlayerSession](#) o [CreatePlayerSessions](#) con un ID di sessione di gioco.

Una richiesta di sessione utente deve includere un ID giocatore univoco. Per ulteriori informazioni, consulta [Genera giocatore IDs](#). Dopo aver ricevuto la richiesta, Amazon GameLift Servers verifica che la sessione di gioco accetti nuovi giocatori e che siano disponibili slot per giocatori. In caso di successo, Amazon GameLift Servers riserva uno slot per il giocatore, crea la nuova sessione di gioco e restituisce un [PlayerSession](#) oggetto.

Una sessione di gioco può includere una serie di dati personalizzati del giocatore. Questi dati vengono memorizzati nell'oggetto della sessione del giocatore appena creato. Amazon GameLift Servers passa questo oggetto al server di gioco quando il giocatore si connette direttamente alla

sessione di gioco. Quando richiedi sessioni con più giocatori, associa ogni stringa di dati del giocatore a un ID giocatore.

2. Connect a una sessione di gioco

Aggiungi codice al servizio di backend per recuperare l'`PlayerSession` oggetto (ad esempio chiamando [DescribePlayerSessions\(\)](#)) e passalo nuovamente al client di gioco. Questo oggetto contiene il nome DNS, l'indirizzo IP e la porta. Il client di gioco può utilizzare queste informazioni per stabilire una connessione diretta al server.

- È possibile connettersi utilizzando la porta specificata e il nome DNS o l'indirizzo IP assegnato al processo del server.
- Se le tue flotte hanno abilitato la generazione di certificati TLS, connettiti utilizzando il nome e la porta DNS.
- Se il server di gioco convalida le connessioni dei giocatori in entrata, fai riferimento all'ID della sessione del giocatore.

Dopo aver effettuato la connessione, il client di gioco e il processo server comunicano direttamente senza coinvolgere Amazon GameLift Servers. Il server mantiene la comunicazione con Amazon GameLift Servers cui segnalare lo stato della connessione del giocatore, lo stato di salute e altro ancora.

Se il server di gioco convalida i giocatori in arrivo, verifica che l'ID della sessione del giocatore corrisponda a uno slot riservato nella sessione di gioco e accetta o nega la connessione del giocatore. Quando il giocatore si disconnette, il processo del server segnala l'interruzione della connessione.

Passa dati di gioco personalizzati a una sessione di gioco

Il tuo client di gioco può trasferire dati a una sessione di gioco. Quando crei una sessione di gioco, puoi includere nella richiesta un set di proprietà di gioco (coppie chiave-valore) o dati della sessione di gioco (valori stringa). Puoi anche aggiornare una sessione di gioco esistente con dati di gioco nuovi o aggiornati. Questi dati vengono trasmessi al processo del server di gioco che ospita la sessione di gioco, per essere utilizzati dal codice del server di gioco. Non puoi eliminare le proprietà del gioco.

Ad esempio, supponiamo che il gioco definisca i livelli di difficoltà: `NoviceEasy`, `Intermediate`, e `Expert`. Un giocatore sceglie di partecipare a una `Easy` partita. Il tuo client di gioco, tramite il servizio di backend, richiede una nuova sessione di gioco con la seguente proprietà di gioco:

`{"Key": "Difficulty", "Value": "Easy"}` In risposta, Amazon GameLift Servers richiede a un server di gioco disponibile di avviare una nuova sessione di gioco e passa l'oggetto. `GameSession` Il processo del server di gioco utilizza la proprietà di gioco fornita per impostare il livello di difficoltà della sessione di gioco.

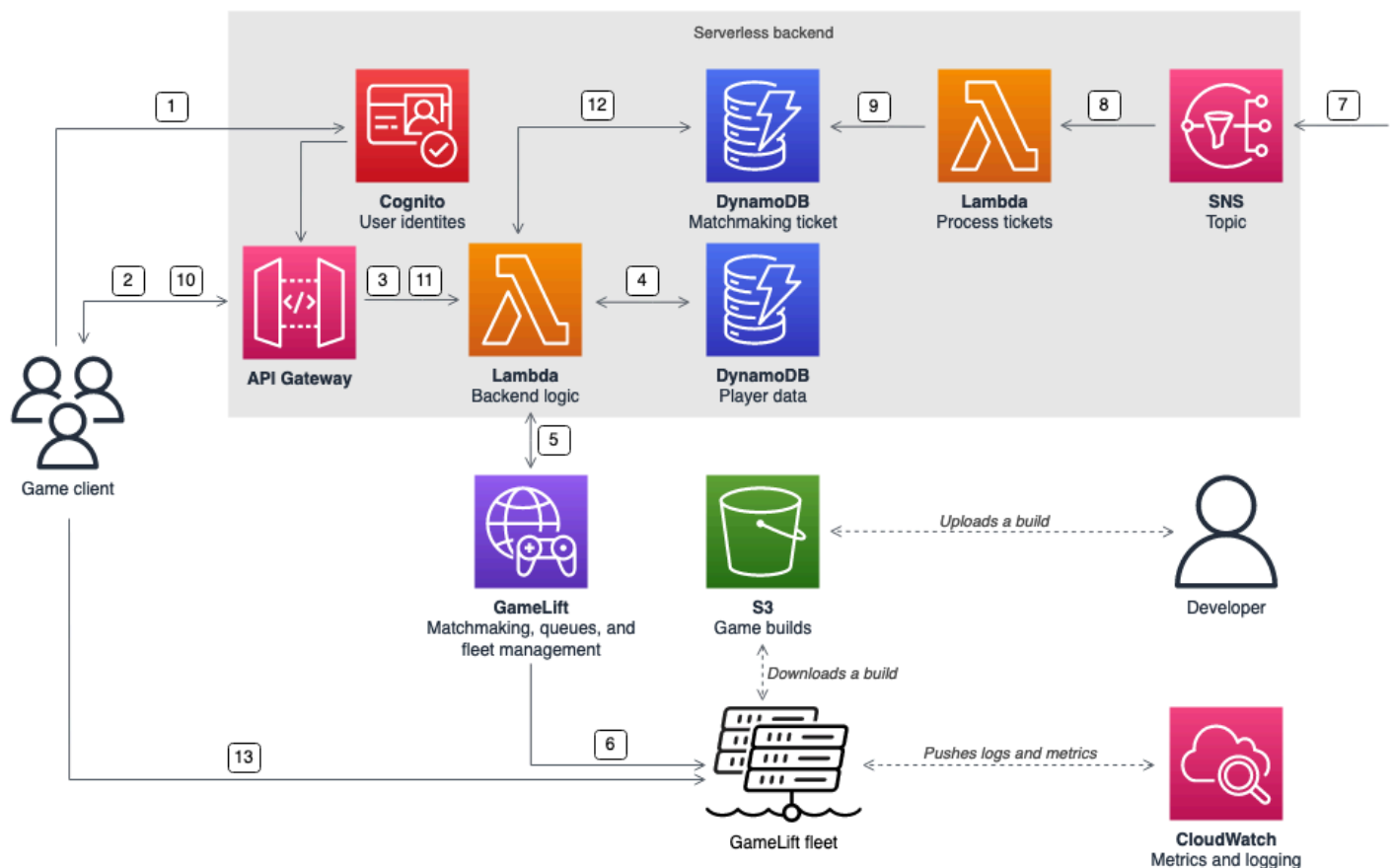
Ulteriori informazioni

- [GameProperty tipo di dati](#)
- [SearchGameSessions\(\) esempi](#)
- [UpdateGameSession GameProperties parametro \(\)](#)

Server di sessione di gioco autonomi con un backend senza server

Utilizzando un'architettura di servizio client serverless, il backend può visualizzare lo stato dei ticket di matchmaking da un database altamente scalabile anziché accedere direttamente all'API. Amazon GameLift Servers

Il diagramma seguente mostra un backend serverless costruito con Servizi AWS che abbina i giocatori alle partite in esecuzione su flotte. Amazon GameLift Servers L'elenco seguente fornisce una descrizione per ogni callout numerato nel diagramma. Per provare questo esempio, vedi Hosting di giochi [basato su sessioni multigiocatore](#) on on. AWS GitHub



1. Il client di gioco richiede un'identità utente Amazon Cognito da un pool di identità Amazon Cognito.
2. Il client di gioco riceve credenziali di accesso temporanee e richiede una sessione di gioco tramite un'API Amazon API Gateway.
3. API Gateway richiama una AWS Lambda funzione.
4. La funzione Lambda richiede i dati del giocatore da una tabella NoSQL di Amazon DynamoDB. La funzione fornisce l'identità di Amazon Cognito nei dati del contesto della richiesta.
5. La funzione Lambda richiede una corrispondenza tramite Amazon GameLift Servers FlexMatch matchmaking.
6. FlexMatchabbina un gruppo di giocatori con una latenza adeguata, e quindi richiede il posizionamento della sessione di gioco tramite una coda. Amazon GameLift Servers La coda ha flotte con una o più Regione AWS postazioni al suo interno.
7. Dopo aver effettuato Amazon GameLift Servers la sessione in una delle sedi della flotta, Amazon GameLift Servers invia una notifica di evento a un argomento di Amazon Simple Notification Service (Amazon SNS).
8. Una funzione Lambda riceve l'evento Amazon SNS e lo elabora.

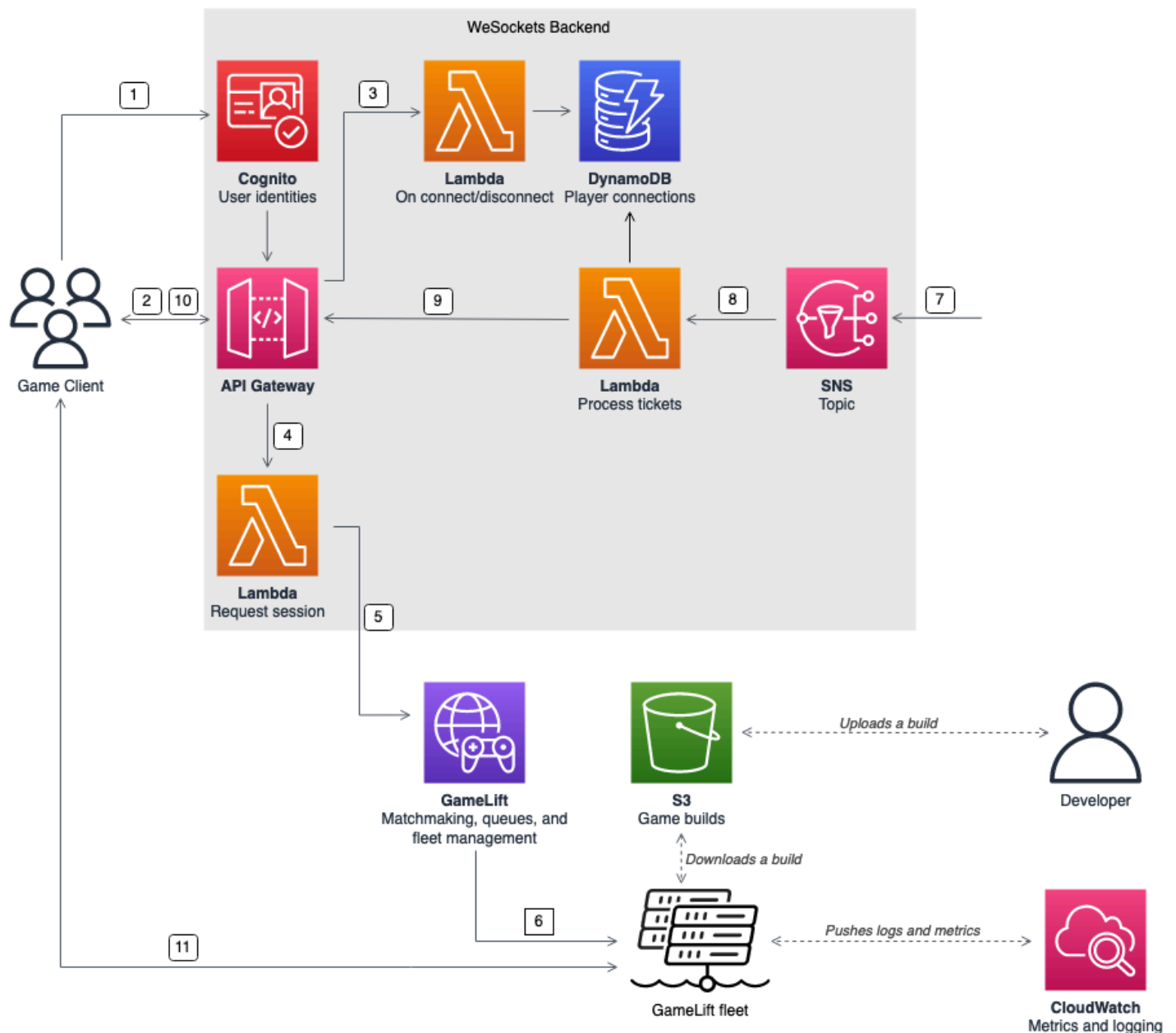
9. Se il ticket di matchmaking è un MatchmakingSucceeded evento, la funzione Lambda scrive il risultato, insieme alla porta e all'indirizzo IP del server di gioco, in una tabella DynamoDB.
- 10 Il client di gioco invia una richiesta firmata ad API Gateway per visualizzare lo stato del ticket di matchmaking in un intervallo specifico.
- 11 API Gateway utilizza una funzione Lambda che verifica lo stato del ticket di matchmaking.
- 12 La funzione Lambda controlla la tabella DynamoDB per verificare se il ticket ha esito positivo. Se ha avuto successo, la funzione invia la porta e l'indirizzo IP del server di gioco, insieme all'ID della sessione del giocatore, al client. Se il ticket non è andato a buon fine, la funzione invia una risposta per verificare che la partita non sia ancora pronta.
- 13 Il client di gioco si connette al server di gioco tramite TCP o UDP utilizzando la porta e l'indirizzo IP forniti dal servizio di backend. Il client di gioco invia quindi l'ID di sessione del giocatore al server di gioco, che quindi convalida l'ID utilizzando l'SDK del server per Amazon GameLift Servers.

Server di sessione di gioco autonomi con un backend basato WebSocket

Utilizzando un'architettura WebSocket basata su Amazon API Gateway, puoi effettuare richieste di matchmaking WebSockets e inviare notifiche push per il completamento del matchmaking utilizzando messaggi avviati dal server. Questa architettura migliora le prestazioni grazie alla comunicazione bidirezionale tra il client e il server.

Per ulteriori informazioni sull'utilizzo di API Gateway WebSock APIs, consulta [Working with WebSocket APIs](#).

Il diagramma seguente mostra un'architettura di backend WebSocket basata su API Gateway e altri Servizi AWS per abbinare i giocatori ai giochi in esecuzione su Amazon GameLift Servers flotte. L'elenco seguente fornisce una descrizione per ogni callout numerato nel diagramma.



1. Il client di gioco richiede un'identità utente Amazon Cognito da un pool di identità Amazon Cognito.
2. Il client di gioco firma una WebSocket connessione a un'API Gateway API con le credenziali di Amazon Cognito.
3. API Gateway richiama una AWS Lambda funzione sulla connessione. La funzione memorizza le informazioni di connessione in una tabella Amazon DynamoDB.
4. Il client di gioco invia un messaggio a una funzione Lambda, tramite l'API API Gateway tramite la WebSocket connessione, per richiedere una sessione.

5. Una funzione Lambda riceve il messaggio e quindi richiede una corrispondenza tramite Amazon GameLift Servers FlexMatch matchmaking.
6. Dopo FlexMatch le partite, un gruppo di giocatori, FlexMatch richiede il posizionamento di una sessione di gioco tramite una Amazon GameLift Servers coda.
7. Dopo aver effettuato Amazon GameLift Servers la sessione in una delle sedi della flotta, Amazon GameLift Servers invia una notifica di evento a un argomento di Amazon Simple Notification Service (Amazon SNS).
8. Una funzione Lambda riceve l'evento Amazon SNS e lo elabora.
9. Se il ticket di matchmaking è un MatchmakingSucceeded evento, la funzione Lambda richiede la connessione corretta del giocatore da DynamoDB. La funzione invia quindi un messaggio al client di gioco tramite l'API Gateway API tramite la WebSocket connessione. In questa architettura, il client di gioco non verifica attivamente lo stato del matchmaking.
- 10 Il client di gioco riceve la porta e l'indirizzo IP del server di gioco, insieme all'ID della sessione del giocatore, tramite la WebSocket connessione.
- 11 Il client di gioco si connette al server di gioco tramite TCP o UDP utilizzando la porta e l'indirizzo IP forniti dal servizio di backend. Il client di gioco invia inoltre l'ID di sessione del giocatore al server di gioco, che quindi convalida l'ID utilizzando l'SDK del server per. Amazon GameLift Servers

Implementa flotte di hosting per Amazon GameLift Servers

L'implementazione delle risorse di hosting implica la creazione e la configurazione dell'infrastruttura di elaborazione su cui verranno eseguiti i server di gioco. Amazon GameLift Servers offre diversi tipi di flotta per soddisfare diverse esigenze di hosting, da Cloud AWS risorse completamente gestite a soluzioni ibride che combinano infrastruttura cloud e locale.

Scegli il tipo di flotta più adatto alle tue esigenze in termini di costi, controllo, scalabilità e distribuzione geografica. Puoi anche combinare più tipi di flotta in un'unica soluzione di hosting per ottimizzare diversi scenari o popolazioni di giocatori.

Caratteristiche della flotta

Una Amazon GameLift Servers flotta è una raccolta di risorse informatiche che gestiscono i server di gioco e ospitano sessioni di gioco per i giocatori. Le flotte possono variare in base al tipo di risorse di elaborazione utilizzate e al modo in cui viene gestita la flotta. Le dimensioni di una flotta, ovvero il numero di sessioni di gioco e di giocatori che può supportare, dipendono dal numero di risorse di elaborazione fornite. Tutte le flotte hanno le seguenti caratteristiche Amazon GameLift Servers:

- I processi del server di gioco eseguiti su tutte le flotte sono integrati con l'SDK del server Amazon GameLift Servers e comunicano con il Amazon GameLift Servers servizio allo stesso modo. I server di gioco segnalano la loro disponibilità a ospitare sessioni di gioco e giocatori, rispondono alle richieste di avvio o interruzione delle sessioni di gioco e altre interazioni.
- Amazon GameLift Servers gestisce il posizionamento delle sessioni di gioco per tutte le flotte allo stesso modo. Amazon GameLift Servers tiene traccia dello stato del server di gioco di una flotta e sceglie tra i server di gioco disponibili per ospitare una nuova sessione di gioco. Questo processo viene utilizzato se il gioco colloca le sessioni di gioco su una singola flotta o utilizza una [coda di sessioni di gioco](#) per bilanciare l'hosting tra più flotte. Con una coda, puoi anche personalizzare le decisioni di posizionamento tenendo conto di fattori come il costo delle risorse e la latenza.
- Tutte le flotte supportano l'uso di un FlexMatch matchmaker in collaborazione con una coda per il posizionamento delle sessioni di gioco. Il Amazon GameLift Servers servizio riceve le richieste di partite dei giocatori, forma le partite e le passa alla coda delle sessioni di gioco per trovare i server di gioco disponibili.
- Amazon GameLift Servers raccoglie un'ampia gamma di metriche sulla flotta. Queste includono metriche di stato per computer e processi del server, nonché metriche di utilizzo per le sessioni di gioco e l'attività dei giocatori. Consulta l'elenco completo delle metriche disponibili all'indirizzo. [Monitora Amazon GameLift Servers con Amazon CloudWatch](#)

Argomenti

- [Guida decisionale: scegli un'opzione di hosting](#)
- [Amazon GameLift Servers EC2 flotte gestite](#)
- [Amazon GameLift Servers flotte di container gestite](#)
- [Amazon GameLift Servers Flotte ovunque](#)
- [Crea una soluzione di hosting ibrida](#)

Guida decisionale: scegli un'opzione di hosting

Amazon GameLift Servers offre diverse opzioni di hosting, ognuna progettata per diversi scenari e requisiti di hosting. Questa guida ti aiuta a scegliere il tipo di flotta in base alle esigenze tecniche del gioco, alle preferenze operative e ai vincoli aziendali. Per descrizioni di alto livello, consulta. [Amazon GameLift Servers opzioni di hosting di giochi](#)

Confronto della flotta

La tabella seguente mette a confronto le caratteristiche chiave di ciascuna opzione di hosting:

Opzione di hosting	Ideale per	Livello di gestione	Vantaggi principali
Gestito EC2	Hosting di produzione e con AWS gestione completa	Servizio completamente gestito	Scalabilità automatica, portata globale, sovraccarico operativo minimo
Contenitori gestiti	Applicazioni containerizzate che richiedono l'orchestrazione	Servizio completamente gestito	Orchestrazione dei container, efficienza delle risorse, implementazione moderna
Ovunque	Ambienti locali, ibridi o di sviluppo	Autogestito	Controllo dell'hardware, requisiti di conformità, ottimizzazione dei costi

Fattori decisionali chiave

Considera questi fattori quando scegli la tua opzione di hosting:

Complessità operativa

Quanta gestione dell'infrastruttura vuoi gestire? Le flotte gestite richiedono un sovraccarico operativo minimo, mentre le flotte Anywhere offrono il pieno controllo ma richiedono una maggiore gestione.

Requisiti di scalabilità

Hai bisogno di un ridimensionamento automatico in base alla domanda dei giocatori? Le flotte gestite offrono l'auto-scaling integrato, mentre le flotte Anywhere richiedono la gestione manuale della capacità.

Distribuzione geografica

Dove si trovano i tuoi giocatori? Le flotte gestite possono essere distribuite in più AWS regioni a livello globale, mentre le flotte Anywhere dipendono dalle ubicazioni dell'infrastruttura aziendale.

Conformità e sicurezza

Hai requisiti di conformità specifici o hai bisogno di conservare i dati in locale? Le flotte Anywhere ti consentono di mantenere il pieno controllo sull'infrastruttura e sulla posizione dei dati.

Considerazioni sui costi

Qual è la tua strategia di ottimizzazione del budget e dei costi? Quando confronti le opzioni, considera sia i costi dell'infrastruttura che i costi operativi.

Soluzioni ibride

Non devi scegliere un solo tipo di flotta per la tua soluzione di hosting di giochi. Molti giochi utilizzano un approccio ibrido che combina diversi tipi di flotta per ottimizzare diversi scenari:

- Sviluppo e test: utilizza le flotte Anywhere per lo sviluppo e i test, quindi implementale su flotte gestite per la produzione.
- Ottimizzazione regionale: utilizza flotte gestite in aree Regioni AWS ad alta densità di giocatori e flotte Anywhere nelle regioni in cui è presente un'infrastruttura esistente.
- Ottimizzazione dei costi: utilizza flotte gestite per la capacità di base e flotte Anywhere per sovraccarichi o carichi di lavoro specifici.

Amazon GameLift Servers EC2 flotte gestite

Amazon GameLift Servers EC2 le flotte gestite forniscono risorse basate sul cloud per l'hosting di produzione. Con una flotta gestita, ottieni la flessibilità, la sicurezza e l'affidabilità di Cloud AWS risorse ottimizzate per l'hosting di giochi multiplayer. Amazon GameLift Servers fornisce solidi strumenti di gestione degli host.

Una EC2 flotta gestita è un insieme di istanze Amazon Elastic Compute Cloud (Amazon EC2) che Amazon GameLift Servers possiede e opera in base alla tua configurazione. Queste istanze si trovano fisicamente nelle Local Zones Regioni AWS o supportate. Quando crei una flotta, scegli un tipo di EC2 istanza che soddisfi i requisiti del server di gioco in termini di potenza di calcolo, memoria, archiviazione e funzionalità di rete.

All'avvio di ogni istanza del parco istanze, Amazon GameLift Servers distribuisce la build del server di gioco con l'ambiente di runtime richiesto. L'ambiente di runtime utilizza l'ultima versione di Amazon Machine Image (AMI) disponibile al momento della creazione del parco macchine. Tutte le istanze del parco istanze utilizzano la stessa versione AMI.

Note

Come best practice, consigliamo di sostituire le flotte ogni 30 giorni per mantenere un ambiente sicuro e di up-to-date runtime per i server di gioco ospitati. Ciò richiede la creazione di una nuova flotta e la migrazione del traffico di giocatori verso di essa. Per ulteriori informazioni, consulta [Best practice relative alla sicurezza di Amazon GameLift Servers](#).

Dopo aver installato l'ambiente di runtime e aver creato il server di gioco su un'istanza, Amazon GameLift Servers inizia ad avviare i processi del server di gioco. Ogni processo del server di gioco stabilisce una connessione al Amazon GameLift Servers servizio, segnala la disponibilità a ospitare una sessione di gioco e inizia a comunicare lo stato di salute. Amazon GameLift Servers può quindi richiedere al processo del server di avviare una sessione di gioco.

Oltre alla distribuzione della flotta, Amazon GameLift Servers gestisce le seguenti attività di gestione dell'host in modo da non dover:

- Monitora lo stato di tutti i computer del parco computer e sostituisce i computer obsoleti o non integri.
- Gestisce l'autenticazione per la comunicazione tra i processi del server e il servizio. Amazon GameLift Servers
- Avvia e arresta automaticamente i processi del server di gioco su ogni computer, in base alla configurazione di runtime.
- Offre strumenti di ridimensionamento automatico per regolare la capacità della flotta in modo dinamico per soddisfare la domanda dei giocatori.
- Riporta le metriche delle prestazioni per le istanze del parco istanze. EC2

Consulta questi argomenti su come configurare e mantenere flotte gestite EC2 :

- [Roadmap di sviluppo per l'hosting con EC2 Amazon GameLift Servers gestito](#)
- [Crea una EC2 flotta Amazon GameLift Servers gestita](#)
- [Scalabilità della capacità di hosting di giochi con Amazon GameLift Servers](#)

- [Personalizzazioni delle risorse di hosting](#)
- [Aggiornamento di una configurazione del Amazon GameLift Servers parco veicoli](#)

Flusso di lavoro gestito EC2 per la creazione di

Per le flotte gestite, Amazon GameLift Servers configura la risorsa del parco veicoli e distribuisce anche una serie di risorse di elaborazione con il software del server di gioco installato e funzionante. Quando il flusso di lavoro di creazione è completo e ha esito positivo, la flotta dispone di un' EC2 istanza attiva nella regione di origine della flotta e una in ciascuna delle sedi remote della flotta. Tutte le istanze dispongono di giochi pronti per ospitare sessioni di gioco.

1. Amazon GameLift Servers crea la risorsa del parco veicoli nella regione di origine della flotta e imposta la capacità desiderata in ogni sede su una (1) istanza. Lo stato della flotta e dell'ubicazione è impostato su Nuovo.
2. Amazon GameLift Servers inizia a scrivere gli eventi nel registro degli eventi della flotta.
3. Amazon GameLift Servers imposta lo stato della flotta su Downloading e inizia a preparare il software del server di gioco per l'implementazione.
 - a. Ottiene la build del server di gioco caricata ed estrae i file compressi.
 - b. Esegue gli script di installazione, se forniti.
 - c. Imposta lo stato del parco macchine su Convalida e inizia a verificare che non si siano verificati errori durante il download e l'installazione dei file di build.
4. Amazon GameLift Servers imposta lo stato del parco macchine su Building, configura l'hardware del parco macchine e alloca un' EC2 istanza per ogni istanza del parco macchine.
5. Amazon GameLift Servers imposta lo stato della flotta su Attivazione. Avvia un processo del server di gioco su ogni istanza (in base alle istruzioni di runtime della flotta) e verifica la connettività tra la build e il Amazon GameLift Servers servizio.
6. Quando i processi del server di gioco su ciascuna istanza stabiliscono una connessione e segnalano la disponibilità a ospitare sessioni di gioco, Amazon GameLift Servers imposta gli stati della flotta e della posizione su Attivo. A questo punto, la flotta è considerata pronta per ospitare sessioni di gioco.

Crea una EC2 flotta Amazon GameLift Servers gestita

Questo argomento descrive come creare un EC2 parco veicoli Amazon GameLift Servers gestito. Le flotte gestite utilizzano istanze di calcolo Amazon Elastic Compute Cloud EC2 (Amazon) ottimizzate

per l'hosting di giochi multiplayer. Puoi creare flotte gestite che distribuiscono i computer a livello globale verso le Local Regioni AWS Zones supportate da Amazon GameLift Servers

Quando si crea una nuova risorsa gestita per il EC2 parco veicoli, si avvia immediatamente la prima fase di creazione del parco veicoli. La creazione gestita della flotta passa attraverso diverse fasi: Amazon GameLift Servers distribuisce un' EC2 istanza, installa un ambiente di runtime e il server di gioco si basa sull'istanza e inizia ad avviare i server di gioco. A seconda dell'ambiente di runtime richiesto dalla build del server di gioco, Amazon GameLift Servers distribuisce la versione più recente di Amazon Machine Image (AMI) al momento della creazione della flotta (e tutte le future istanze del parco istanze utilizzeranno la stessa versione). Puoi monitorare lo stato di un parco macchine nella console o utilizzando (). AWS Command Line Interface AWS CLI Quando una flotta è pronta per ospitare sessioni di gioco, lo stato cambia in. ACTIVE Per ulteriori informazioni sulla creazione gestita della flotta, consulta questi argomenti:

Note

Come best practice, consigliamo di sostituire le flotte ogni 30 giorni per mantenere un ambiente sicuro e di up-to-date runtime per i server di gioco ospitati. Ciò richiede la creazione di una nuova flotta e la migrazione del traffico di giocatori verso di essa. Per ulteriori informazioni, consulta [Best practice relative alla sicurezza di Amazon GameLift Servers](#).

Per creare una flotta gestita EC2

Usa la Amazon GameLift Servers console o AWS Command Line Interface (AWS CLI) per creare una EC2 flotta gestita.

Console

Nella [Amazon GameLift Serversconsole](#), utilizza il riquadro di navigazione per aprire la pagina Gestito EC2, flotte. Scegli Crea EC2 flotta gestita per avviare il flusso di lavoro di creazione della flotta.

Fase 1 Definisci i dettagli EC2 della flotta gestita

Per una configurazione minima della flotta:

- Fornisci un nome per la flotta.

- Scegli il tipo binario: Build. Scegli una build dall'elenco delle build di server su cui hai precedentemente caricato. Amazon GameLift Servers
- Salta le sezioni Dettagli aggiuntivi e Tag per utilizzare le impostazioni predefinite.

1. Compila la sezione Dettagli della flotta:

- a. Inserisci un nome per la flotta. I nomi delle flotte sono solo descrittivi e sono utili per ordinare o filtrare un elenco di flotte. Non è necessario che siano unici.
- b. Fornisci una breve descrizione della flotta per includere tutte le informazioni personalizzate che desideri acquisire sulla flotta.
- c. Seleziona Tipo binario, Crea e usa l'elenco a discesa per scegliere una build del server di gioco su cui hai precedentemente caricato il file. Amazon GameLift Servers

2. Imposta dettagli aggiuntivi in base alle esigenze.

- a. Se il file eseguibile del server di gioco deve accedere ad altre AWS risorse del tuo account, specifica un ruolo di istanza IAM con le autorizzazioni necessarie. Per ulteriori informazioni, incluso come autorizzare altre applicazioni lato server (come l' CloudWatch agente), consulta. [Connect il server di gioco Amazon GameLift Servers ospitato ad altre AWS risorse](#) Questa impostazione non può essere modificata dopo aver creato la flotta.

È necessario creare il ruolo prima di creare una flotta che lo utilizzi. Inoltre, per creare una flotta con un ruolo di istanza, AWS l'utente deve disporre dell'PassRole autorizzazione IAM (vedi [Esempi di autorizzazioni IAM per Amazon GameLift Servers](#)).

- b. Attiva l'opzione Genera un certificato TLS per configurare l'autenticazione e la crittografia per il gioco. I client di gioco utilizzano questo certificato per autenticare un server di gioco durante la connessione e crittografare tutte le comunicazioni client/server. Per ogni istanza in un parco dispositivi abilitati per TLS. Amazon GameLift Servers crea anche una nuova voce DNS con il certificato. Questa impostazione non può essere modificata dopo aver creato la flotta.
- c. Amazon GameLift Server mette dati metrici per ogni singola flotta. Se desideri combinare i dati metrici per più flotte, specifica il nome di un gruppo metrico. Utilizza lo stesso nome del gruppo di metriche per tutte le flotte per cui desideri combinare le metriche. Utilizzalo CloudWatch per visualizzare i dati aggregati del gruppo metrico.

3. Aggiungi tag alla risorsa del parco veicoli. Ogni tag è composto da una chiave e da un valore opzionale, entrambi personalizzabili. Assegna tag alle AWS risorse che desideri classificare, ad esempio per scopo, proprietario o ambiente. Scegli Aggiungi nuovo tag per ogni tag che desideri aggiungere.
4. Scegli Avanti per continuare il flusso di lavoro.

Fase 2 Definire i dettagli dell'istanza

In questo passaggio, seleziona il tipo di risorse hardware da utilizzare per i tuoi server di gioco e specifica dove vuoi distribuirle. Scegliendo più sedi, puoi distribuire il server di gioco in una posizione geografica più ampia, in modo da avvicinarli ai giocatori e ridurre al minimo la latenza. Tutte le EC2 istanze di un parco istanze devono essere dello stesso tipo. Non tutti i tipi di istanze sono disponibili in tutte le località.

 Per una configurazione minima del parco veicoli:

- Inizia con la distribuzione solo nella sede principale. Non è necessario aggiungere postazioni remote per lo sviluppo e il test iniziali.
- Imposta il tipo di flotta impostato su «On-Demand». Le flotte Spot richiedono un lavoro di configurazione aggiuntivo.
- Imposta il tipo di istanza su «c5.large». Questo tipo di istanza comunemente usato è disponibile in tutti. Regioni AWS È possibile ottimizzare l'utilizzo dell'hardware in un secondo momento.

1. Nella distribuzione dell'istanza, specifica l'ubicazione e il tipo del parco veicoli.
 - a. Seleziona una o più sedi aggiuntive in cui desideri distribuire le istanze del parco istanze. Queste postazioni remote vengono aggiunte alla sede principale della flotta (che è preselezionata), che è la posizione Regione AWS in cui stai creando questa flotta. È possibile selezionare postazioni remote tra tutte Regioni AWS le Local Zones Amazon GameLift Servers supportate.

Per ulteriori informazioni sulle località supportate, incluso come utilizzare quelle Regione AWS che non sono abilitate [Amazon GameLift Servers sedi di assistenza](#) per impostazione predefinita, consulta Informazioni sull'hosting gestito. Consulta anche Amazon GameLift Servers [le quote relative](#) alle sedi per flotta.

- b. Scegli di utilizzare istanze On-demand o Spot per questo parco istanze. Per ulteriori informazioni sui tipi di parco veicoli, consulta [On-Demand Instances versus Spot Instances](#)
2. Scegli una configurazione Amazon EC2 Instance che soddisfi le tue esigenze e sia disponibile in tutte le località selezionate. Questo elenco viene filtrato in base alla posizione corrente e alle selezioni del tipo di flotta. Puoi filtrarlo ulteriormente in base ad altri fattori, come il tipo di istanza, la famiglia e l'architettura. Dopo aver creato il parco istanze, non puoi modificare il tipo di istanza.

Alcune sedi hanno opzioni di tipo di istanza limitate. Se il tipo di istanza preferito non è disponibile per tutte le sedi, scegli il valore di disponibilità della sede per visualizzare tutti i dettagli. Per ospitare tutte le sedi, potrebbe essere necessario creare flotte separate con diversi tipi di istanze.

Per ulteriori informazioni sulla scelta di un tipo di istanza, consulta [Tipi di istanza](#). Per ulteriori informazioni sulle architetture Amazon EC2 Arm, consulta [Processore AWS Graviton e tipi di istanze Amazon EC2](#). Per un elenco completo dei tipi di istanze supportati da Amazon GameLift Servers, consulta il riferimento API per [EC2InstanceType\(\)](#). `CreateFleet()`

 Note

Le istanze Graviton Arm richiedono una build del server per un'AMI Linux. Per C++ e C# è richiesto Server SDK 5.1.1 o versione successiva. Per Go è richiesto Server SDK 5.0 o versione successiva. Queste istanze non forniscono out-of-the-box supporto per l'installazione di Mono su Amazon Linux 2023 (AL2023) o Amazon Linux 2 (). AL2

3. Scegli Avanti per continuare il flusso di lavoro.

Fase 4 Configurare il runtime

In questo passaggio, descrivi come desideri che ogni istanza del parco istanze esegua il software del server di gioco. Definisci una riga di processo del server distinta per ogni eseguibile da eseguire su un'istanza e decidi quanti processi server eseguire contemporaneamente. Apri le porte su ogni istanza per consentire ai giocatori di connettersi

direttamente ai server di gioco. Puoi aggiornare queste impostazioni della flotta in qualsiasi momento.

 Per una configurazione minima del parco veicoli:

- Definisci un singolo elemento della riga di processo del server per il file eseguibile del server di gioco. Se il server di gioco richiede l'esecuzione di altri processi, crea anche una definizione per ciascuno di questi.
- Utilizza il numero predefinito di processi simultanei (1) per ogni elemento della riga.
- Salta le impostazioni di attivazione della sessione di gioco.
- Specificare un numero di porta singolo.
- Salta le impostazioni delle risorse della sessione di gioco.

1. Crea una configurazione Runtime per istruzioni Amazon GameLift Servers su come eseguire i processi del server su ogni istanza del parco istanze. Puoi modificare la configurazione di runtime di una flotta in qualsiasi momento dopo l'implementazione.
 - a. Inserisci il percorso di lancio di un file eseguibile nella tua build. Nelle istanze di Windows, gli eseguibili dei server di gioco vengono creati in base al percorso. C : \game Sulle istanze Linux, i server di gioco sono compilati su /local/game. Esempi: **C:\game\MyGame\server.exe** o **/local/game/MyGame/server.exe**.
 - b. Inserisci i parametri di avvio opzionali da passare al file eseguibile del gioco. Esempio: **+sv_port 33435 +start_lobby**.
 - c. Specificate il numero di processi simultanei da eseguire su ogni istanza. Per un server di gioco eseguibile, ogni processo può ospitare una sessione di gioco, quindi i processi simultanei determinano il numero di sessioni di gioco che l'istanza può ospitare contemporaneamente.

Controlla le Amazon GameLift Servers [quote](#) sui processi del server per istanza. Queste quote si applicano al totale dei processi simultanei per tutte le configurazioni. Se configuri la flotta in modo da superarle, la flotta non può attivarsi.
2. Usa le impostazioni predefinite per l'attivazione della sessione di gioco o personalizzate per il tuo gioco. Se la configurazione di runtime richiede più processi simultanei del server di gioco per istanza, queste impostazioni determinano la velocità di avvio delle nuove sessioni di gioco.

- a. Imposta l'attivazione massima della sessione di gioco simultanea per limitare il numero di server di gioco su un'istanza che stanno preparando una nuova sessione di gioco. Questa impostazione è utile quando l'avvio di più nuove sessioni di gioco richiede molte risorse e potrebbe influire sulle prestazioni di altre sessioni di gioco in corso.
 - b. Imposta il nuovo timeout di attivazione in modo da indicare il tempo massimo che una nuova sessione di gioco deve impiegare per completare l'attivazione e segnalare che è pronta per ospitare i giocatori. Amazon GameLift Servers interrompe l'attivazione di una sessione di gioco se supera questo valore.
3. Apri le impostazioni delle EC2 porte per consentire al traffico in entrata di accedere ai processi del server del parco macchine. Queste impostazioni non sono necessarie per creare una flotta, ma è necessario configurarle prima che i giocatori possano connettersi alle sessioni di gioco della flotta.

Per ogni impostazione di porta, scegli il tipo di protocollo di trasferimento dati da utilizzare per la comunicazione tra il client di gioco e il server di gioco. Fornisci un intervallo di porte (in formato `nnnnn[-nnnnn]`) e un intervallo di indirizzi IP utilizzando la notazione CIDR (ad esempio, `0.0.0.0/0` che consente l'accesso a chiunque).

Se devi impostare più intervalli non consecutivi, crea più impostazioni di porta.

- a. Attiva o disattiva la politica di protezione di Game Scaling per tutte le istanze del parco istanze. Durante un evento di ridimensionamento, non Amazon GameLift Servers interromperà le istanze protette del parco istanze che ospitano sessioni di gioco attive.
 - b. Imposta un limite massimo per la creazione di risorse se desideri limitare il numero di sessioni di gioco che un giocatore può creare durante un periodo di tempo specificato.
5. Scegli Avanti per continuare il flusso di lavoro.

Passaggio 5: rivedi e crea

Controlla le impostazioni prima di creare la flotta. Sebbene alcune impostazioni possano essere aggiornate in un secondo momento (vedi [Aggiornamento di una configurazione del](#)

[Amazon GameLift Servers parco veicoli](#)), le modifiche alle seguenti impostazioni non sono consentite dopo la creazione del parco veicoli:

- Tipo di elaborazione: non puoi convertire una EC2 flotta gestita in una flotta Anywhere.
- Server binario di gioco: per implementare un aggiornamento alla build del tuo server di gioco, devi creare una nuova flotta.
- Opzioni aggiuntive, tra cui il ruolo dell'istanza e la generazione di certificati TLS.
- Dettagli dell'istanza, incluso il tipo di flotta (Spot o On-Demand) e il tipo di EC2 istanza.

Quando sei pronto per implementare la nuova flotta, scegli Crea. Amazon GameLift Servers avvia immediatamente il processo di attivazione della flotta, assegnando un ID univoco e impostando lo stato della flotta. NEW Tieni traccia dei progressi della flotta dalla pagina Flotte. Visualizza la pagina dei dettagli della flotta e vai alla scheda Eventi.

Puoi modificare la capacità di hosting di una flotta dopo che la flotta ha raggiunto lo stato ATTIVO. Amazon GameLift Servers inizialmente implementa una flotta con una singola istanza in ogni sede del parco macchine e tu modifichi la capacità aggiungendo istanze a ciascuna sede. Per ulteriori informazioni, consulta [Scalabilità della capacità di hosting di giochi con Amazon GameLift Servers](#).

AWS CLI

Usa il [create-fleet](#) comando per creare una flotta di tipi EC2 di calcolo. Amazon GameLift Servers crea la risorsa fleet con l'impostazione predefinita corrente Regione AWS (oppure puoi aggiungere un tag `--region` per specificarne un altro Regione AWS).

Crea una flotta gestita minima

La seguente richiesta di esempio crea una nuova flotta con le impostazioni minime necessarie per implementare una flotta con server di gioco in esecuzione a cui i client di gioco possono connettersi. La nuova flotta presenta le seguenti caratteristiche:

- Specifica una build del server di gioco, che è stata caricata su Amazon GameLift Servers e in READY stato.
- Utilizza istanze On-Demand c5.large con un sistema operativo che corrisponde alla build del gioco selezionata.
- Imposta la sede della flotta us-west-2 e distribuisce Regione AWS le istanze solo in quella regione.

- In base alla configurazione di runtime, ogni computer del parco computer esegue un processo del server di gioco, il che significa che ogni computer può ospitare solo una sessione di gioco alla volta. Il timeout di attivazione della sessione di gioco è impostato sul valore predefinito di 300 secondi e non c'è limite al numero di attivazioni simultanee.
- I giocatori possono connettersi ai server di gioco utilizzando l'impostazione di una singola porta di. 33435
- Tutte le altre funzionalità sono disattivate o utilizzano le impostazioni predefinite.

```
aws gamelift create-fleet \  
  --name MinimalFleet123 \  
  --description "A basic test fleet" \  
  --region us-west-2 \  
  --ec2-instance-type c5.large \  
  --fleet-type ON_DEMAND \  
  --build-id build-1111aaaa-22bb-33cc-44dd-5555eeee66ff \  
  --runtime-configuration "ServerProcesses=[{LaunchPath=C:\game\Bin64.dedicated  
\MultiplayerSampleProjectLauncher_Server.exe, ConcurrentExecutions=10}]" \  
  --ec2-inbound-permissions  
  "FromPort=33435,ToPort=33435,IpRange=0.0.0.0/0,Protocol=UDP"
```

Crea una flotta gestita completamente configurata

La seguente richiesta di esempio crea una flotta di produzione con impostazioni per tutte le funzionalità opzionali. La nuova flotta presenta le seguenti caratteristiche:

- Specifica una build del server di gioco, che è stata caricata su Amazon GameLift Servers e in READY stato.
- Utilizza le istanze On-Demand c5.large con il sistema operativo corrispondente alla build di gioco selezionata.
- Imposta la sede del parco istanze us-west-2 e distribuisce Regione AWS le istanze nella regione di origine e in una postazione remota. sa-east-1
- In base alla configurazione di runtime:
 - Ogni computer del parco computer esegue 10 processi del server di gioco con gli stessi parametri di avvio, il che significa che ogni computer può ospitare fino a 10 sessioni di gioco contemporaneamente.

- Su ogni computer, possono essere attivate solo due sessioni di gioco contemporaneamente. Le sessioni di gioco attivate devono essere pronte a ospitare giocatori entro 300 secondi (5 minuti) o essere interrotte.
- I giocatori possono connettersi ai server di gioco utilizzando una porta compresa nel seguente intervallo. 33435 to 33535
- Genera un certificato TLS per le comunicazioni crittografate tra client di gioco e server.
- In tutte le sessioni di gioco della flotta è attivata la protezione della sessione di gioco.
- I singoli giocatori sono limitati a creare tre nuove sessioni di gioco entro un periodo di 15 minuti.
- Le metriche per questa flotta sono incluse nel gruppo di metriche `AMERfleets`, che (per questo esempio) aggrega le metriche per un gruppo di flotte in Nord, Centro e Sud America.

```
aws gamelift create-fleet \
  --name ProdFleet123 \
  --description "A fully configured prod fleet" \
  --ec2-instance-type c5.large \
  --region us-west-2 \
  --locations "Location=sa-east-1" \
  --fleet-type ON_DEMAND \
  --build-id build-1111aaaa-22bb-33cc-44dd-5555eeee66ff \
  --certificate-configuration "CertificateType=GENERATED" \
  --runtime-configuration "GameSessionActivationTimeoutSeconds=300,
  MaxConcurrentGameSessionActivations=2, ServerProcesses=[{LaunchPath=C:\game
\Bin64.dedicated\MultiplayerSampleProjectLauncher_Server.exe, Parameters=+sv_port
33435 +start_lobby, ConcurrentExecutions=10}]" \
  --new-game-session-protection-policy "FullProtection" \
  --resource-creation-limit-policy "NewGameSessionsPerCreator=3,
PolicyPeriodInMinutes=15" \
  --ec2-inbound-permissions
  "FromPort=33435,ToPort=33535,IpRange=0.0.0.0/0,Protocol=UDP" \
  --metric-groups "AMERfleets"
```

Se la richiesta di creazione della flotta ha esito positivo, Amazon GameLift Servers restituisce un set di attributi della flotta che include le impostazioni di configurazione richieste e un nuovo ID della flotta. Amazon GameLift Servers avvia quindi il processo di attivazione della flotta e imposta lo stato della flotta e gli stati della posizione su Nuovo. Puoi monitorare lo stato del parco istanze e visualizzare altre informazioni tramite questi comandi dell'interfaccia CLI:

- [describe-fleet-events](#)

- [describe-fleet-attributes](#)
- [describe-fleet-capacity](#)
- [describe-fleet-port-settings](#)
- [describe-fleet-utilization](#)
- [describe-runtime-configuration](#)
- [describe-fleet-location-attributes](#)
- [describe-fleet-location-capacity](#)
- [describe-fleet-location-utilization](#)

È possibile modificarne la capacità e altre impostazioni di configurazione in base alle esigenze tramite questi comandi:

- [update-fleet-attributes](#)
- [update-fleet-capacity](#)
- [update-fleet-port-settings](#)
- [update-runtime-configuration](#)
- [create-fleet-locations](#)
- [delete-fleet-locations](#)

Amazon GameLift Servers flotte di container gestite

Amazon GameLift Servers le flotte di container gestite forniscono risorse basate sul cloud per ospitare il software del server di gioco containerizzato. Con una flotta di container gestita, ottieni la flessibilità, la sicurezza e l'affidabilità delle Cloud AWS risorse, ottimizzate per l'hosting di giochi multiplayer. Amazon GameLift Servers fornisce solidi strumenti di gestione degli host.

 **Accelera l'onboarding con questi strumenti per contenitori gestiti:**

- Lo [starter kit per container](#) semplifica l'integrazione e la configurazione della flotta. Aggiunge funzionalità essenziali per la gestione delle sessioni di gioco al server di gioco e utilizza modelli preconfigurati per creare una flotta di container e una pipeline di distribuzione automatizzata per il server di gioco. Dopo l'implementazione, utilizza la Amazon GameLift Servers console e gli strumenti API per monitorare le prestazioni della flotta, gestire le sessioni di gioco e analizzare le metriche.

- Per gli sviluppatori di Unreal Engine o Unity, usa i [Amazon GameLift Servers plugin e il server del motore di gioco SDKs per integrare il tuo server](#) di gioco e creare una flotta di container dall'interno dell'ambiente di sviluppo del tuo motore di gioco. I flussi di lavoro guidati del plug-in ti aiutano a creare una soluzione semplice e veloce con l'hosting basato su cloud utilizzando contenitori gestiti. Puoi basarti su queste basi per creare una soluzione di hosting personalizzata per il tuo gioco.

Una flotta di container gestita è un insieme di istanze Amazon Elastic Compute Cloud (Amazon EC2) che eseguono Linux, che Amazon GameLift Servers possiede e opera in base alla tua configurazione. Queste istanze si trovano fisicamente nelle Local Zones Regioni AWS o supportate. Quando crei una flotta di container, scegli un tipo di EC2 istanza che soddisfi i requisiti del server di gioco in termini di potenza di calcolo, memoria, archiviazione e funzionalità di rete.

Per una flotta di container gestita, memorizzi le immagini dei container basate su Linux in un repository Amazon Elastic Container Registry (Amazon ECR) e crei una definizione di gruppo di contenitori per descrivere l'architettura dei container. Quando crei una flotta, effettua il Amazon GameLift Servers provisioning di un'istanza della flotta con la versione più recente di Linux Amazon Machine Image (AMI) e utilizza la definizione del gruppo di contenitori per distribuire le immagini dei container. Tutte le istanze di una flotta di container utilizzeranno la stessa versione AMI, anche se aggiorni una definizione di gruppo di container o modifichi un'immagine del contenitore.

Note

Come best practice, consigliamo di sostituire le flotte ogni 30 giorni per mantenere un ambiente sicuro e di up-to-date runtime per i server di gioco ospitati. Ciò richiede la creazione di una nuova flotta e la migrazione del traffico di giocatori verso di essa. Per ulteriori informazioni, consulta [Best practice relative alla sicurezza di Amazon GameLift Servers](#).

Dopo aver distribuito l'istanza containerizzata, i container iniziano ad avviare i processi del server di gioco. Ogni processo del server di gioco stabilisce una connessione al Amazon GameLift Servers servizio, segnala la disponibilità a ospitare una sessione di gioco e inizia a comunicare lo stato di salute. Amazon GameLift Servers può quindi richiedere al processo del server di avviare una sessione di gioco.

Oltre alla distribuzione della flotta, Amazon GameLift Servers gestisce le seguenti attività di gestione dell'host in modo da non dover:

- Monitora lo stato di tutti i container della flotta e sostituisce quelli obsoleti o insalubri.
- Gestisce l'autenticazione per la comunicazione tra i processi del server e il servizio. Amazon GameLift Servers
- Offre strumenti di scalabilità automatica che regolano la capacità della flotta in modo dinamico per soddisfare la domanda dei giocatori.
- Riporta le metriche delle prestazioni per le EC2 istanze, i container e i processi server della flotta.

Consulta questi argomenti su come configurare e gestire flotte di container gestite:

- [Roadmap di sviluppo per l'hosting con contenitori Amazon GameLift Servers gestiti](#)
- [Crea una flotta di container Amazon GameLift Servers gestita](#)
- [Personalizza una flotta di Amazon GameLift Servers container](#)
- [Scalabilità della capacità di hosting di giochi con Amazon GameLift Servers](#)
- [Aggiorna una flotta di container Amazon GameLift Servers gestita](#)

Come funzionano i contenitori Amazon GameLift Servers

Amazon GameLift Servers le flotte di container sono progettate per offrirti flessibilità nel modo in cui distribuisce e ridimensiona le tue applicazioni containerizzate. Utilizza Amazon Elastic Container Service (Amazon ECS) per gestire la distribuzione e l'esecuzione delle attività per le tue flotte. Amazon GameLift Servers Questo argomento descrive gli elementi strutturali di base per far funzionare i container su una flotta Amazon GameLift Servers gestita, illustra le architetture comuni e delinea alcuni concetti fondamentali.

 **Accelera l'onboarding con questi strumenti per container gestiti:**

- Lo [starter kit per container](#) semplifica l'integrazione e la configurazione della flotta. Aggiunge funzionalità essenziali per la gestione delle sessioni di gioco al server di gioco e utilizza modelli preconfigurati per creare una flotta di container e una pipeline di distribuzione automatizzata per il server di gioco. Dopo l'implementazione, utilizza la Amazon GameLift Servers console e gli strumenti API per monitorare le prestazioni della flotta, gestire le sessioni di gioco e analizzare le metriche.
- Per gli sviluppatori di Unreal Engine o Unity, usa i [Amazon GameLift Servers plugin](#) per integrare il tuo server di gioco e creare una flotta di container dall'interno dell'ambiente di sviluppo del tuo motore di gioco. I flussi di lavoro guidati del plug-in ti aiutano a creare una

soluzione semplice e veloce con l'hosting basato su cloud utilizzando contenitori gestiti. Quindi sfrutta queste basi per creare una soluzione di hosting personalizzata per il tuo gioco.

Componenti della flotta di container

Parco istanze

Una flotta di container è una raccolta di istanze Amazon EC2 per ospitare i tuoi server di gioco containerizzati. Queste istanze sono gestite da te per tuo conto. Amazon GameLift Servers Quando crei una flotta, configuri il modo in cui l'architettura del container con il software del server di gioco viene distribuita a ciascuna istanza della flotta. Puoi creare una flotta di container con istanze in una o più località geografiche. Puoi utilizzare strumenti di Amazon GameLift Servers scalabilità per scalare automaticamente la capacità di una flotta di container di ospitare sessioni di gioco e giocatori.

Istanza

Un'istanza Amazon EC2 è il server virtuale che fornisce capacità di elaborazione per l'hosting di giochi. Con Amazon GameLift Servers, puoi scegliere tra una vasta gamma di tipi di istanze. Ogni tipo di istanza offre una combinazione diversa di CPU, memoria, storage e capacità di rete.

Quando crei una flotta di container, Amazon GameLift Servers distribuisce i container in base al tipo di istanza scelto e alla configurazione della flotta. Ogni istanza della flotta distribuita è identica e esegue il software del server di gioco containerizzato nello stesso modo. Il numero di istanze in un parco istanze determina le dimensioni del parco istanze e la capacità di hosting dei giochi.

Gruppo di contenitori

Amazon GameLift Servers utilizza il concetto di gruppo di contenitori per descrivere e gestire un insieme di contenitori. Un gruppo di contenitori è simile a un contenitore «task» o «pod». All'interno di ciascun gruppo di contenitori, è possibile definire il comportamento dei contenitori, impostare le dipendenze e condividere le risorse di CPU e memoria disponibili.

Ogni istanza della flotta può avere i seguenti tipi di gruppi di contenitori:

- Un gruppo di contenitori di server di gioco gestisce i contenitori che eseguono l'applicazione del server di gioco e il software di supporto. Una flotta di container deve disporre di uno di questo tipo di gruppo di container per ospitare sessioni di gioco e giocatori. Un gruppo di container di server di gioco può essere replicato su un'istanza della flotta. Il numero di repliche di gruppi di

server di gioco per istanza del parco istanze dipende dai requisiti di elaborazione del software e dalle risorse di calcolo disponibili sull'istanza.

- Un gruppo di container per istanza, facoltativo, ti dà la possibilità di eseguire software aggiuntivo su ogni istanza del parco istanze. Sono utili per eseguire servizi in background o programmi di utilità, ad esempio per il monitoraggio. Il software del server di gioco non dipende direttamente dai processi di un gruppo di istanze. In ogni istanza del parco veicoli viene distribuita una sola copia di un gruppo di container per istanza.

Ogni gruppo di container in una flotta di container ha un container designato come «essenziale». Un container essenziale determina il ciclo di vita di un gruppo di container. Se il contenitore essenziale si guasta, l'intero gruppo di contenitori si riavvia.

Contenitore

Il contenitore è l'elemento più basilare di un'architettura basata su contenitori. Include un'immagine del contenitore con eseguibili software e file dipendenti. Definisci un contenitore per configurare il modo in cui il software viene eseguito e con cui interagisce. Amazon GameLift Servers

Amazon GameLift Servers definisce due tipi di contenitori:

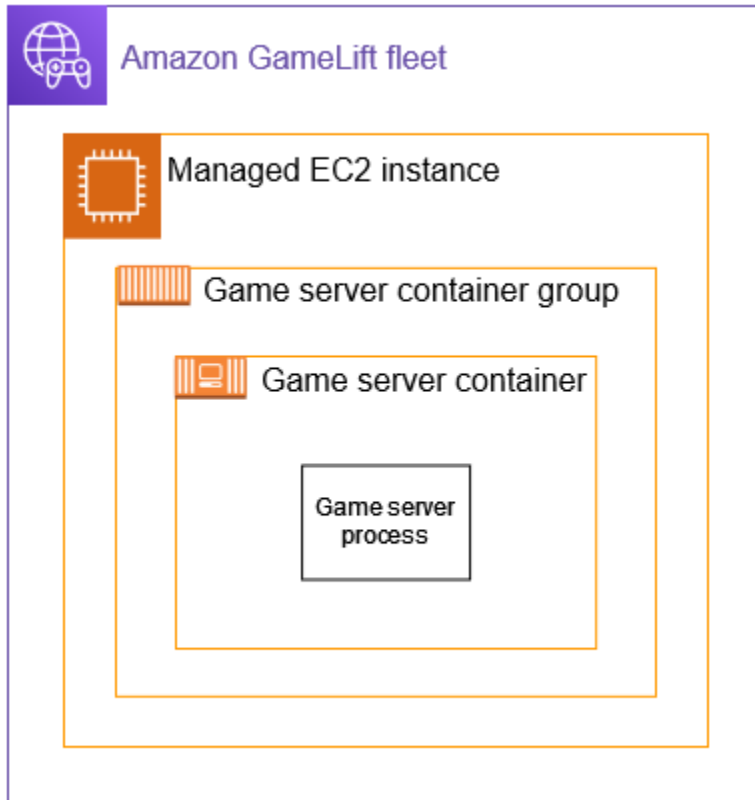
- Un contenitore per server di gioco include tutto ciò di cui hai bisogno per eseguire i processi del server di gioco e ospitare sessioni di gioco per i giocatori. Include la build del server di gioco e il software dipendente. Definisci un contenitore di server di gioco per il gruppo di contenitori di server di gioco di una flotta. Il contenitore del server di gioco viene automaticamente considerato essenziale per il gruppo di contenitori.
- Un contenitore di supporto esegue software aggiuntivo per supportare il server di gioco. È simile al concetto di contenitore «sidecar». Ti dà la possibilità di eseguire e scalare il software di supporto insieme ai server di gioco, gestendolo però come contenitori separati. In un gruppo di contenitori di server di gioco, puoi definire zero o più contenitori di supporto. In un gruppo di contenitori per istanza, tutti i contenitori sono contenitori di supporto. Qualsiasi contenitore di supporto può essere considerato essenziale.

Calcolo

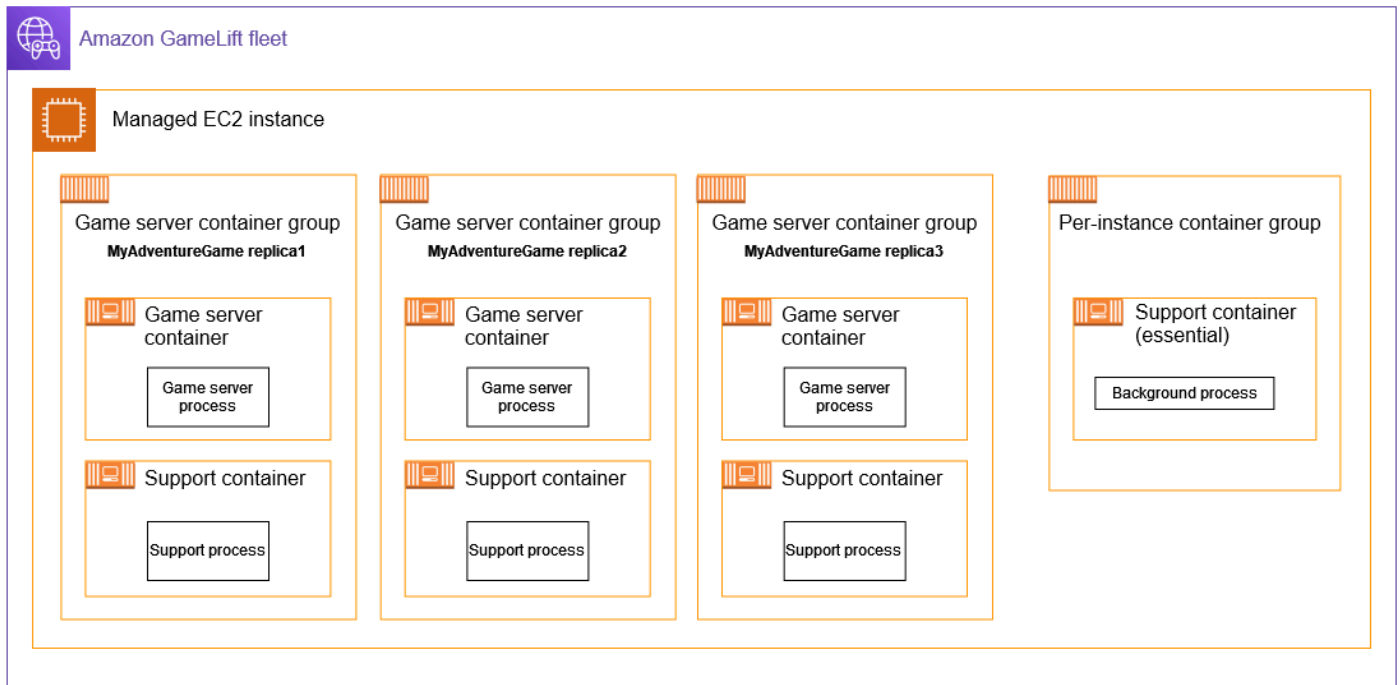
Un calcolo rappresenta una copia di un gruppo di contenitori di server di gioco su un'istanza della flotta.

Architetture comuni

Il diagramma seguente illustra la struttura più semplice della flotta di container. In questa struttura, ogni istanza della flotta mantiene una copia del gruppo di contenitori del server di gioco. Il gruppo di contenitori ha un unico contenitore di server di gioco che esegue un processo del server di gioco. In questo esempio, la flotta di container è configurata per collocare una copia del gruppo di contenitori del server di gioco per istanza. Con questa architettura, ogni istanza esegue un processo del server di gioco.



Questo secondo diagramma illustra un'architettura di flotte di container più complessa. In questa struttura, la flotta dispone sia di un gruppo di container di server di gioco che di un gruppo di container per istanza. Il gruppo di contenitori di server di gioco ha contenitori separati per il processo del server di gioco e un processo di supporto. Il parco macchine è configurato per posizionare tre copie del gruppo di contenitori del server di gioco su ciascuna istanza del parco macchine. Il gruppo di contenitori per istanza non viene mai replicato. In questo esempio, la flotta di container è configurata per collocare tre copie del gruppo di contenitori del server di gioco per istanza. Con questa architettura, ogni istanza esegue tre processi del server di gioco.



Caratteristiche principali

Questa sezione riassume come Amazon GameLift Servers implementa alcuni concetti di base sui container. Per istruzioni su come lavorare con le flotte di container, consulta gli argomenti pertinenti di questa guida.

Aggiornamenti attivi della flotta

I container gestiti forniscono un supporto avanzato per aiutarti a gestire il ciclo di vita del software ospitato e dell'architettura dei container. Puoi aggiornare le definizioni dei container, incluse le immagini dei container, e implementare modifiche alle flotte esistenti. Questa funzionalità semplifica e velocizza l'iterazione delle modifiche ai container durante lo sviluppo. Fornisce inoltre funzionalità che consentono di creare, distribuire e tenere traccia degli aggiornamenti delle versioni del software nel tempo. Queste funzionalità includono:

- Gestisci gli aggiornamenti e il controllo delle versioni delle definizioni dei gruppi di contenitori. È possibile aggiornare quasi tutte le proprietà di una definizione di gruppo di contenitori, incluse le immagini dei contenitori e le impostazioni di configurazione. Ogni volta che aggiorni un contenitore, assegna Amazon GameLift Servers automaticamente un numero di versione all'aggiornamento e, per impostazione predefinita, mantiene tutte le versioni. È possibile accedere a qualsiasi versione specifica ed eliminare le versioni desiderate. Quando si crea una flotta di container, è possibile specificare la definizione del gruppo di container e la versione da distribuire.

- Aggiorna le flotte di container esistenti con nuove definizioni di gruppi di contenitori e impostazioni di configurazione. Puoi distribuire gli aggiornamenti dei container alle flotte già distribuite nelle istanze del parco veicoli. Puoi monitorare lo stato delle distribuzioni degli aggiornamenti in ogni sede del parco veicoli utilizzando Console di gestione AWS o l' AWS SDK e la CLI.
- Configura come desideri che gli aggiornamenti del parco macchine vengano distribuiti in un parco macchine attivo.
- Protezione delle sessioni di gioco. Scegli di proteggere le istanze della flotta con sessioni di gioco attive fino al termine delle sessioni di gioco (distribuzione sicura). Oppure scegli di sostituire le istanze della flotta indipendentemente dall'attività della sessione di gioco (distribuzione non sicura). Utilizza implementazioni non sicure durante le fasi di sviluppo e test per ridurre i tempi di implementazione.
- Percentuale minima di salute. Specificate la percentuale di attività integre che desiderate mantenere durante la distribuzione. Questa funzionalità ti consente di decidere quante istanze del parco istanze subire l'impatto durante una distribuzione. Un valore basso dà priorità alla velocità di implementazione, mentre un valore alto assicura che la disponibilità dei server di gioco rimanga elevata per tutta la durata dell'implementazione.
- Strategia di fallimento dell'implementazione. Decidi quali azioni intraprendere se una distribuzione fallisce. Un errore di distribuzione significa che alcuni contenitori aggiornati non hanno superato i controlli di stato e sono considerati compromessi. È possibile impostare le distribuzioni per ripristinare automaticamente tutte le istanze del parco istanze allo stato di distribuzione precedente. In alternativa, puoi scegliere di mantenere alcune delle istanze del parco istanze danneggiate da utilizzare nel debug.

La possibilità di aggiornare le flotte attive è molto utile quando desideri implementare un aggiornamento al software del tuo server di gioco. Dopo aver creato una nuova immagine del container per il server di gioco, la sua implementazione è un processo in due fasi: in primo luogo, aggiorna la definizione del gruppo di container con la nuova immagine e, in secondo luogo, aggiorna la flotta di container. Amazon GameLift Servers gestisce tutte le altre attività in base alle esigenze.

Imballaggio in contenitori

Quando si sviluppa la struttura dei container per l'implementazione in una flotta di container, un obiettivo comune è ottimizzare l'uso della potenza di calcolo disponibile. Per raggiungere questo obiettivo, devi raggruppare il maggior numero possibile di gruppi di container di server di gioco in ciascuna istanza della flotta.

Amazon GameLift Serversti aiuta a farlo calcolando il numero massimo di gruppi di contenitori di server di gioco per istanza, in base alle seguenti informazioni:

- Il tipo di istanza del parco istanze e le relative risorse di vCPU e memoria.
- I requisiti di vCPU e memoria per tutti i contenitori nel gruppo di contenitori del server di gioco.

I requisiti di vCPU e memoria per tutti i contenitori in un gruppo di contenitori per istanza, se presente.

Quando si crea una flotta di container, è possibile utilizzare il massimo calcolato o specificare il numero desiderato. Come procedura ottimale, pianifica di sperimentare con il software del server di gioco containerizzato per determinare i requisiti di risorse per prestazioni ottimali del server di gioco.

Dimensionamento della capacità

La capacità della flotta misura il numero di sessioni di gioco che una flotta può ospitare contemporaneamente. Puoi anche misurare la capacità in base al numero di giocatori simultanei che una flotta può supportare. Per aumentare o diminuire la capacità di hosting di una flotta, aggiungi o rimuovi istanze della flotta.

Una flotta di container è configurata per eseguire un numero specifico di processi simultanei dei server di gioco su ciascuna istanza della flotta. (Puoi calcolarlo in base (1) ai gruppi di contenitori dei server di gioco per istanza e (2) al numero di processi dei server di gioco eseguiti in ciascun gruppo di contenitori.) Il numero di server di gioco simultanei per istanza indica l'impatto dell'aggiunta o della rimozione di ciascuna istanza del parco istanze. Ad esempio, se la tua flotta di container esegue 1 processo di server di gioco in ogni gruppo di container di server di gioco e ogni istanza della flotta contiene 100 gruppi di contenitori di server di gioco, aumenti o diminuisci la capacità della tua flotta di ospitare sessioni di gioco simultanee con incrementi di 100. Se ogni sessione di gioco ha 10 slot per giocatori, puoi aumentare o diminuire la capacità della flotta di ospitare giocatori con incrementi di 1000.

Con le flotte di container, puoi utilizzare uno qualsiasi dei metodi di scalabilità della capacità forniti da Amazon GameLift Servers. Ciò include:

- Imposta manualmente la capacità della flotta impostando il numero di istanze della flotta desiderato.
- Imposta il ridimensionamento automatico scegliendo come target un buffer desiderato di istanze disponibili (tracciamento del target). Questo metodo mantiene automaticamente una certa quantità

di risorse di hosting inattive in modo che i giocatori in arrivo possano iniziare rapidamente a giocare. Man mano che la domanda dei giocatori aumenta o diminuisce, la dimensione di questo buffer viene continuamente adattata.

- Imposta il ridimensionamento automatico con regole di ridimensionamento personalizzate (funzionalità avanzata). Questo metodo ti consente di scalare in base alle metriche della flotta che scegli.

Connessioni di gioco client/server

Con le flotte Amazon GameLift Servers gestite, i client di gioco si connettono direttamente ai server di gioco ospitati sul cloud. Quando un client di gioco chiede di partecipare a una partita, Amazon GameLift Servers trova una sessione di gioco e fornisce le informazioni di connessione (IP e porta) al client di gioco. Puoi controllare l'accesso esterno alle istanze della flotta aprendo determinati intervalli di porte (autorizzazioni in entrata) per la flotta. Le autorizzazioni in entrata determinano quali porte sono aperte al traffico in entrata. È possibile chiudere rapidamente tutte le porte, limitarle a poche o aprire tutte le porte.

Le flotte di container gestite richiedono un'impostazione aggiuntiva che consenta l'accesso ai processi in esecuzione in un container. Quando si crea una definizione di contenitore, si specifica un set di porte, una per ogni processo che richiede una connessione. Questo include:

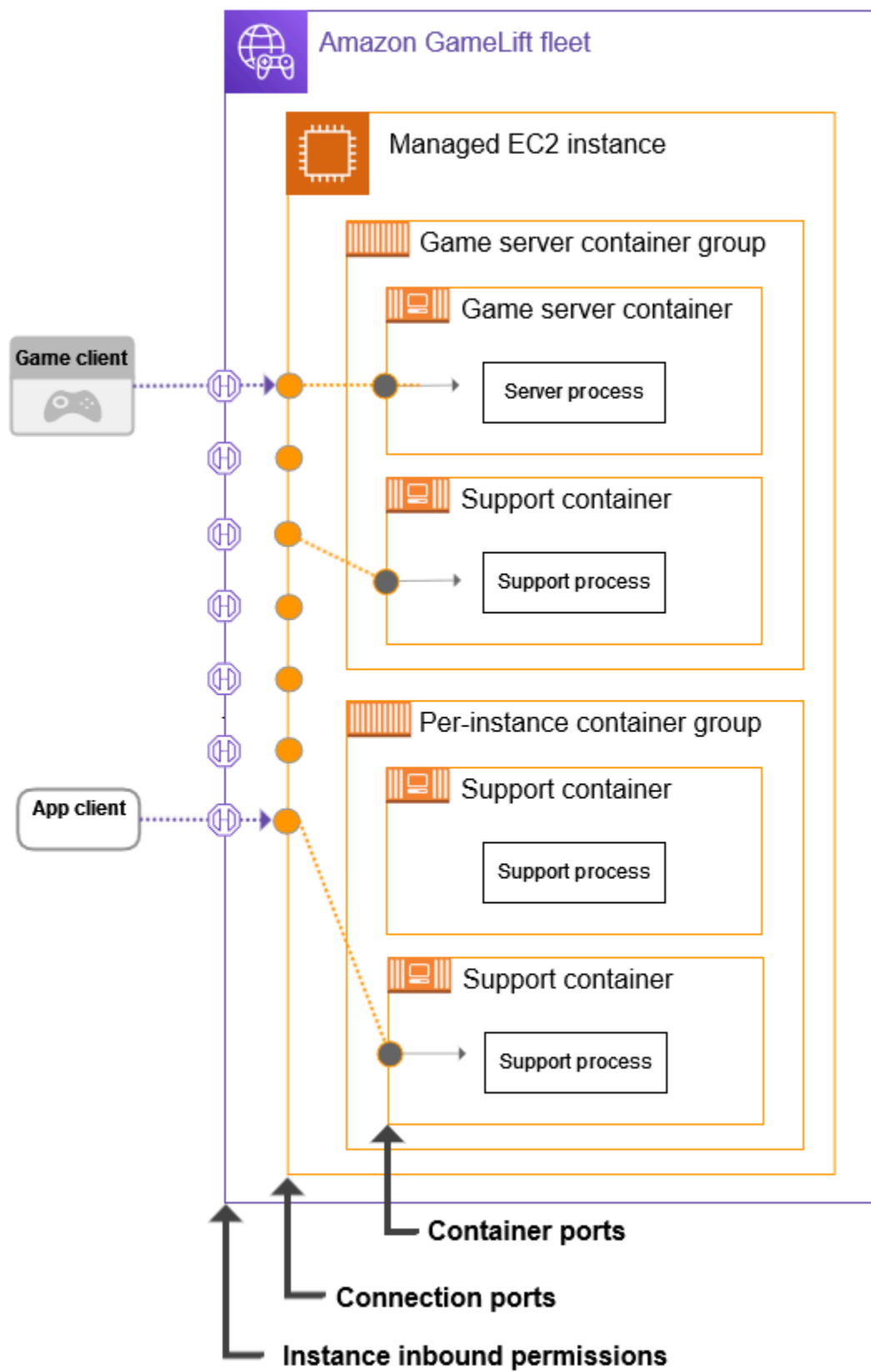
- Tutti i processi del server di gioco che verranno eseguiti contemporaneamente nel contenitore del server di gioco. Tutti i processi del server di gioco devono consentire ai client di gioco di connettersi per partecipare a una sessione di gioco.
- Qualsiasi processo in un contenitore di supporto a cui deve connettersi una fonte esterna. Ad esempio, puoi connetterti in remoto a un'app di test.

Quando configuri le impostazioni della porta container rivolta verso l'interno, le Amazon GameLift Servers utilizza per calcolare le autorizzazioni in entrata rivolte all'esterno a cui possono connettersi i client di gioco e altre applicazioni. Amazon GameLift Servers gestisce anche la mappatura tra le autorizzazioni in entrata e le singole porte container che consente ai giocatori di accedere a una sessione di gioco in un contenitore. Questa mappatura interna fornisce un livello di sicurezza proteggendo i server di gioco dall'accesso diretto alle porte dei container. Hai la possibilità di personalizzare le impostazioni delle porte rivolte verso l'esterno di una flotta in base alle esigenze. Per ulteriori informazioni sull'impostazione manuale dei porti della flotta di container, consulta.

[Configurare le connessioni di rete](#)

Puoi modificare le impostazioni dei porti di una flotta di container in qualsiasi momento. Questa modifica richiede l'implementazione di un aggiornamento della flotta.

Il diagramma seguente illustra il ruolo delle connessioni portuali all'interno di una flotta di container. Come illustrato, si impostano le porte sui singoli container e Amazon GameLift Servers si utilizzano queste informazioni per configurare un numero sufficiente di porte sull'istanza della flotta da mappare a ciascun porto container. Sia le autorizzazioni in ingresso delle istanze rivolte verso l'esterno che le porte di connessione rivolte verso l'interno vengono calcolate in base al tuo parco macchine, a meno che tu non scelga di impostarle manualmente. Amazon GameLift Servers



Registrazione dei container

Nelle flotte di container gestite, i flussi di output standard (e di errore standard) vengono acquisiti per tutti i container. Ciò include i registri delle sessioni di gioco del server di gioco. Puoi configurare una flotta di container per utilizzare una delle diverse opzioni per gestire i flussi di output:

- Salva l'output del contenitore come flusso di CloudWatch log di Amazon. Ogni flusso di log fa riferimento all'ID della flotta e al container. Se si sceglie questa opzione di registrazione per il parco veicoli, si specifica un gruppo di CloudWatch log che organizza tutti i flussi di log del parco veicoli. È quindi possibile utilizzare CloudWatch le funzionalità per cercare e analizzare i dati di registro in base alle esigenze.
- Salva l'output del contenitore in un bucket di storage Amazon Simple Storage Service (Amazon S3). Puoi visualizzare, condividere o scaricare il contenuto in base alle tue esigenze.
- Disattiva la registrazione. In questo scenario, l'output del contenitore non viene salvato.

Amazon GameLift Servers invia i dati di registro dalle flotte di container gestite ai CloudWatch servizi Amazon S3 del tuo account. AWS Per visualizzare i tuoi dati, utilizza questo Console di gestione AWS o altri strumenti accedendo al tuo AWS account e lavorando con i singoli servizi. Estendi l'accesso limitato Amazon GameLift Servers per intraprendere queste azioni creando un ruolo di servizio per le tue flotte di container.

Puoi modificare la configurazione di registrazione di una flotta di container in qualsiasi momento. Questa modifica richiede l'implementazione di un aggiornamento del parco veicoli.

Le flotte di container e l'agente Amazon GameLift Servers

Un'architettura di container di uso comune esegue un singolo processo in ogni contenitore. In una flotta di Amazon GameLift Servers container, il gruppo di contenitori di server di gioco dispone di un contenitore di server di gioco, che esegue un processo del server di gioco. Con questa architettura, Amazon GameLift Servers gestisce il ciclo di vita del singolo processo del server di gioco in ogni gruppo di contenitori di server di gioco su un'istanza della flotta.

Se scegli di creare un'architettura container che esegua più processi di server di gioco in ogni gruppo di contenitori di server di gioco, hai bisogno di un modo per gestire il ciclo di vita di tutti i processi. Ciò include attività come l'avvio, l'arresto e la sostituzione dei processi in base alle esigenze, la gestione del numero desiderato di processi da eseguire contemporaneamente e la gestione degli stati di errore.

È possibile scegliere di utilizzare l'Amazon GameLift Servers agente per queste attività. Per una flotta di container, l'agente implementa istruzioni di runtime che specificano quali eseguibili eseguire (e quanti), forniscono i parametri di avvio e stabiliscono regole per l'attivazione del server di gioco. Ad esempio, le istruzioni di runtime potrebbero indicare all'agente di mantenere dieci processi del server di gioco per l'uso in produzione e un processo del server di gioco con parametri di avvio speciali per i test.

Per utilizzare l'agente con le flotte di container, aggiungete l'agente all'immagine del contenitore e includete un set di istruzioni di runtime. Per ulteriori informazioni sull'agente, consulta [Collabora con l'Amazon GameLift Servers agente](#).

Crea una flotta di container Amazon GameLift Servers gestita

Crea una flotta di container Amazon GameLift Servers gestita per distribuire e ospitare il tuo server di gioco containerizzato nel cloud. AWS Quando crei una flotta di container, specifica le definizioni dei gruppi di container che specificano una o più immagini di container (almeno una che includa la build del server di gioco) e le impostazioni di configurazione.

Quando crei una nuova risorsa gestita per la flotta di container, inizi immediatamente la prima fase della creazione della flotta. La creazione gestita della flotta passa attraverso diverse fasi: esegue il Amazon GameLift Servers provisioning di un'istanza EC2, installa un ambiente di runtime, distribuisce i gruppi di container sull'istanza e inizia ad avviare i processi del server di gioco. A seconda dell'ambiente di runtime richiesto dalla build del server di gioco, Amazon GameLift Servers distribuisce la versione più recente di Amazon Machine Image (AMI) al momento della creazione della flotta (e tutte le istanze future del parco istanze utilizzeranno la stessa versione). Puoi monitorare lo stato di una flotta nella console o utilizzando AWS Command Line Interface (AWS CLI). Quando una flotta è pronta per ospitare sessioni di gioco, lo stato cambia in ACTIVE. Per assistenza in caso di problemi con la creazione del parco istanze, consulta [Debug dei problemi del parco istanze di Amazon GameLift Servers](#).

Puoi scegliere di creare una flotta di container vuota e quindi aggiungere o aggiornare le definizioni dei gruppi di container della flotta in un secondo momento. Se crei una flotta senza una definizione di gruppo di container, la flotta non raggiungerà lo stato attivo.

Note

Come best practice, consigliamo di sostituire le flotte ogni 30 giorni per mantenere un ambiente sicuro e di up-to-date runtime per i server di gioco ospitati. Ciò richiede la creazione

di una nuova flotta e la migrazione del traffico di giocatori verso di essa. Per ulteriori informazioni, consulta [Best practice relative alla sicurezza di Amazon GameLift Servers](#).

Usa la [Amazon GameLift Serversconsole](#) o il AWS Command Line Interface (AWS CLI) per creare una flotta di container.

Console

Nella [Amazon GameLift Serversconsole](#), seleziona il Regione AWS punto in cui desideri creare la flotta. Le definizioni dei gruppi di container devono trovarsi nella stessa regione in cui desideri creare la flotta.

Apri la barra di navigazione a sinistra della console e scegli Contenitori gestiti: flotte. Nella pagina Flotte, scegli Crea flotta di container.

Fase 1: Definisci i dettagli della flotta di container gestita

1. Nella sezione Dettagli della flotta di container, inserisci una descrizione della flotta.
2. Specificate un ruolo IAM per la flotta. Questo ruolo dispone delle autorizzazioni necessarie per gestire la flotta di container per tuo conto. Amazon GameLift Servers Per informazioni sulla creazione del ruolo di servizio richiesto, consulta [Configurare un ruolo di servizio IAM per Amazon GameLift Servers](#).
3. Scegliete un'opzione di configurazione del registro. L' CloudWatch opzione è selezionata per impostazione predefinita. Fornisci le informazioni richieste in base all'opzione selezionata.
4. Aggiungi gruppi di container alla flotta. Questa fase è opzionale. Puoi scegliere di creare una flotta senza un gruppo di container con un piano per aggiungerli in un secondo momento. Una flotta senza gruppi di container non distribuirà alcuna istanza della flotta e non può ancora ospitare giochi, ma la risorsa della flotta viene creata.
 - Seleziona una definizione del gruppo di contenitori del server di gioco. Facoltativamente, specifica la versione della definizione che desideri distribuire. Se non si specifica il numero di versione, utilizza Amazon GameLift Servers automaticamente la versione più recente.
 - Facoltativamente, aggiungi una definizione e una versione del gruppo di contenitori per istanza. Se non specifichi il numero di versione, utilizza Amazon GameLift Servers automaticamente la versione più recente.
5. Nei dettagli aggiuntivi, puoi impostare alcune personalizzazioni opzionali. Nessuna di queste impostazioni è richiesta per creare la flotta di container.

Fase 2: Definizione dei dettagli dell'istanza

1. In Distribuzione delle istanze, seleziona una o più postazioni remote in cui distribuire le istanze. La regione di origine viene selezionata automaticamente (è la regione in cui stai creando la flotta). Se selezioni sedi aggiuntive, le istanze della flotta vengono distribuite anche in queste località.

Important

Per utilizzare le regioni che non sono abilitate per impostazione predefinita, abilitale nelle tue. Account AWS

- Le flotte con regioni non abilitate che hai creato prima del 28 febbraio 2022 non sono interessate da questo requisito.
- Per creare nuove flotte con più sedi o aggiornare le flotte con più sedi esistenti, abilita innanzitutto le Regioni o le Local Zone che scegli di utilizzare.

[Per ulteriori informazioni sulle regioni che non sono abilitate per impostazione predefinita e su come abilitarle, consulta Gestione in. Regioni AWS](#)[Riferimenti generali di AWS](#) Vedi [Guida introduttiva a Local Zones](#) nella AWS Local Zones User Guide.

2. Seleziona una configurazione di istanza per il parco istanze. La console calcola automaticamente la vCPU e la memoria minime richieste (in base ai limiti totali impostati per ogni gruppo di contenitori). Filtra l'elenco completo dei tipi di istanze disponibili in base ai requisiti di risorse e alle posizioni inserite. Se necessario, puoi aggiungere filtri aggiuntivi.

Per ulteriori informazioni sulla scelta del tipo di istanza, consulta [Configura una flotta di container](#). La dimensione del tipo di istanza scelto influirà sul modo in cui i gruppi di container dei server di gioco vengono impacchettati in ciascuna istanza della flotta. A seconda della scelta, valuta la possibilità di rivedere le impostazioni relative ai gruppi di contenitori dei server di gioco desiderati per ogni istanza.

Fase 4: Revisione e creazione

- Rivedi le impostazioni di configurazione del tuo parco veicoli.

Puoi aggiornare i metadati e la configurazione della flotta in qualsiasi momento, indipendentemente dallo stato della flotta. Per ulteriori informazioni, consulta [Aggiornamento di una configurazione del Amazon GameLift Servers parco veicoli](#). Puoi aggiornare la capacità della flotta dopo che la flotta ha raggiunto lo stato ATTIVO. Per ulteriori informazioni, consulta [Scalabilità della capacità di hosting di giochi con Amazon GameLift Servers](#). Puoi anche aggiungere o rimuovere postazioni remote.

Al termine della revisione, scegli Crea.

Se la richiesta ha esito positivo, la console visualizza la pagina di dettaglio della nuova risorsa del parco veicoli. Inizialmente lo stato è NEW, all'Amazon GameLift Servers inizio del processo di creazione della flotta. È possibile monitorare lo stato del nuovo parco istanze nella pagina Fleets (Parchi istanze). Una flotta è pronta per ospitare sessioni di gioco quando raggiunge lo stato ACTIVE.

AWS CLI

Per creare una flotta di container con AWS CLI, apri una finestra della riga di comando e usa il `create-container-fleet` comando. Per ulteriori informazioni su questo comando, vedere [create-container-fleet](#) nella Guida ai AWS CLI comandi.

La `create-container-fleet` richiesta di esempio mostrata di seguito crea una nuova flotta di container con le seguenti caratteristiche:

- `ContainerGroupsConfiguration` Specifica solo una definizione di gruppo di contenitori per server di gioco: `MyAdventureGameContainerGroup`. Il numero di gruppi di contenitori di server di gioco che verranno distribuiti su ciascuna istanza della flotta viene calcolato da Amazon GameLift Servers
- Per impostazione predefinita, la flotta utilizza istanze `c5.large` On-Demand.
- Per impostazione predefinita, la flotta apre una serie di porte di connessione e porte di autorizzazione in entrata, come calcolato da Amazon GameLift Servers. Distribuisce gruppi di container nelle seguenti posizioni:

```
aws gamelift create-container-fleet \  
  --fleet-role-arn arn:aws:iam::MyAccount:role/MyContainersRole \  
  --fleet-name MyGameFleet
```

```
--game-server-container-group-definition-name "rn:aws:gamelift:us-west-2:111122223333:containergroupdefinition/MyAdventureGameContainerGroup:2" \
```

Se la richiesta di creazione della flotta ha esito positivo, Amazon GameLift Servers restituisce un set di attributi della flotta che include le impostazioni di configurazione richieste e un nuovo ID della flotta di container. Amazon GameLift Servers quindi imposta lo stato della flotta e lo stato dell'ubicazione su Nuovo e avvia il processo di attivazione della flotta. Puoi monitorare lo stato del parco istanze e visualizzare altre informazioni tramite questi comandi dell'interfaccia CLI:

- [describe-fleet-events](#)
- [describe-container-fleet](#)
- [describe-fleet-capacity](#)
- [describe-fleet-port-settings](#)
- [describe-fleet-utilization](#)
- [describe-fleet-location-capacity](#)
- [describe-fleet-location-utilization](#)

È possibile modificarne la capacità e altre impostazioni di configurazione in base alle esigenze tramite questi comandi:

- [update-container-fleet](#)
- [update-fleet-capacity](#)
- [update-fleet-port-settings](#)
- [create-fleet-locations](#)
- [delete-fleet-locations](#)

Creare una definizione di gruppo di container per una flotta di Amazon GameLift Servers container

Una definizione di gruppo di container descrive come distribuire le applicazioni di server di gioco containerizzate in una flotta di container. È un modello che indica Amazon GameLift Servers quali immagini di container distribuire nella flotta e come eseguirle. Quando crei una flotta di container, specifichi le definizioni dei gruppi di container da distribuire nella flotta. Per ulteriori informazioni sui gruppi di container, consulta [Componenti della flotta di container](#).

Prima di iniziare

Suggerimenti su cosa fare prima di iniziare a creare una definizione di gruppo di contenitori:

- Finalizza le immagini dei container e trasferiscile in un repository Amazon Elastic Container Registry (Amazon ECR) nello stesso in Regione AWS cui intendi creare il gruppo di contenitori. Amazon GameLift Servers acquisisce un'istantanea di ogni immagine al momento della creazione della definizione del gruppo di contenitori e la utilizza durante la distribuzione in una flotta di container. Per informazioni, consulta [Crea un'immagine del contenitore per Amazon GameLift Servers](#).
- Crea le definizioni dei contenitori come file JSON. Una definizione di gruppo di contenitori include una o più definizioni di contenitori. È possibile utilizzare i file JSON se si crea una definizione di gruppo di contenitori utilizzando AWS CLI for Amazon GameLift Servers
- Verifica che il tuo AWS utente disponga delle autorizzazioni IAM per accedere al repository Amazon ECR. Per informazioni, consulta [Esempi di autorizzazioni IAM per Amazon GameLift Servers](#).

Crea una definizione di gruppo di contenitori per server di gioco

Un gruppo di contenitori di server di gioco esegue il software del server di gioco. Un gruppo di contenitori di server di gioco ha un contenitore di server di gioco, che esegue l'eseguibile del server di gioco. Può anche avere uno o più contenitori di supporto per eseguire software aggiuntivo a supporto del server di gioco. (A volte vengono definiti contenitori «sidecar».)

Questo argomento descrive come creare una semplice definizione di gruppo di contenitori di server di gioco utilizzando la Amazon GameLift Servers console o gli AWS strumenti CLI. Per informazioni più dettagliate sulle funzionalità opzionali, consulta [Personalizza una flotta di Amazon GameLift Servers container](#).

Note

È possibile modificare la maggior parte delle definizioni e delle impostazioni dei gruppi di contenitori dopo averle create. Se apporti modifiche alla definizione di un contenitore, Amazon GameLift Servers acquisisce una nuova istantanea delle immagini del contenitore aggiornate.

Per creare una semplice definizione di gruppo di contenitori per server di gioco:

Le seguenti istruzioni descrivono come creare una definizione di gruppo di contenitori con i parametri minimi richiesti e utilizzando i valori Amazon GameLift Servers predefiniti.

Console

Nella [Amazon GameLift Servers console](#), seleziona il Regione AWS punto in cui desideri creare il gruppo di contenitori.

Apri la barra di navigazione a sinistra della console e scegli Contenitori gestiti: definizioni di gruppo. Nella pagina di definizione dei gruppi di contenitori, scegli Crea definizione di gruppo.

Fase 1: Definire i dettagli della definizione del gruppo di contenitori

1. Immettere un nome per la definizione del gruppo di contenitori. Il nome deve essere univoco per la regione Account AWS and.
2. Seleziona il tipo di gruppo di contenitori del server di gioco.
3. Per Limite di memoria totale, inserisci le risorse di memoria massime da rendere disponibili per tutti i contenitori del gruppo di contenitori. Per informazioni sul calcolo di questo valore, consulta [Imposta i limiti delle risorse](#).
4. Per Limite totale di vCPU, inserisci la potenza di calcolo massima da rendere disponibile per tutti i contenitori del gruppo di contenitori. Per informazioni sul calcolo di questo valore, consulta. [Imposta i limiti delle risorse](#)

Fase 2: Aggiungere le definizioni dei contenitori

Un gruppo di contenitori di server di gioco ha almeno un contenitore di server di gioco. Nella console, la prima definizione di contenitore che crei è il contenitore del server di gioco. Questo passaggio descrive come definire le impostazioni minime richieste per la definizione di un contenitore del server di gioco.

1. Inserisci il nome della definizione del contenitore. Ogni contenitore definito per il gruppo deve avere un valore di nome univoco.
2. Collegati a un'immagine del contenitore con la build del tuo server di gioco. Inserisci l'URI dell'immagine Amazon ECR per l'immagine di un contenitore in un repository pubblico o privato. Puoi utilizzare uno dei seguenti formati:
 - Solo URI dell'immagine: [Account AWS].dkr.ecr.[Regione AWS].amazonaws.com/[repository ID]

- URI dell'immagine + digest: [Account AWS].dkr.ecr.[Regione AWS].amazonaws.com/[repository ID]@[digest]
 - Tag URI + dell'immagine: [Account AWS].dkr.ecr.[Regione AWS].amazonaws.com/[repository ID]:[tag]
3. Specificate la versione Amazon GameLift Servers Server SDK utilizzata dalla build del server di gioco. Per una flotta di container, questo valore deve essere 5.2.0 o superiore.
 4. In Intervallo di porte del container interno, imposta il protocollo e definisci un intervallo di porte. La dimensione dell'intervallo deve essere maggiore del numero di processi simultanei del server di gioco che verranno eseguiti in questo contenitore. Se il contenitore del server di gioco esegue solo un processo server per contenitore, questo intervallo di porte richiede solo poche porte. Per ulteriori dettagli, consultare [Configurare le connessioni di rete](#).
 5. Aggiungi altri contenitori se necessario per eseguire software di supporto aggiuntivo. I contenitori aggiuntivi vengono designati automaticamente come contenitori di supporto. Un gruppo di contenitori di server di gioco può avere solo un contenitore di server di gioco e fino a otto contenitori di supporto. Fornisci le seguenti impostazioni minime richieste:
 - Nome della definizione del contenitore
 - URI dell'immagine ECR.
 - Porte interne del contenitore (includetele solo se il contenitore ha processi che richiedono l'accesso alla rete).

Fase 3: Configurare le dipendenze

- Se la definizione del gruppo di contenitori include più di un contenitore, puoi facoltativamente impostare le dipendenze tra i contenitori. Per ulteriori informazioni, consulta [Imposta le dipendenze del container](#).

Fase 3: Revisione e creazione

1. Rivedi tutte le impostazioni di definizione del gruppo di contenitori. Usa Modifica per apportare modifiche a qualsiasi sezione, inclusa ciascuna delle definizioni dei contenitori per il gruppo.
2. Al termine della revisione, scegli Crea.

Se la richiesta ha esito positivo, la console visualizza la pagina di dettaglio per la nuova risorsa di definizione del gruppo di contenitori. Inizialmente lo stato è `COPYING`: Amazon GameLift Servers inizia a scattare istantanee di tutte le immagini del contenitore per il gruppo. Al termine di questa fase, lo stato della definizione del gruppo di contenitori cambia in `READY`. Una definizione di gruppo di container deve essere `READY` attiva prima di poter creare una flotta di container con essa.

AWS CLI

Quando utilizzi la AWS CLI per creare una definizione di gruppo di contenitori, mantieni le configurazioni della definizione del contenitore in un file separato. JSON Puoi fare riferimento al file nel tuo comando CLI. Vedi alcuni esempi [Crea un file di definizione del contenitore JSON](#) di schema.

Crea una definizione di gruppo di contenitori

Per creare una nuova definizione di gruppo di contenitori, utilizzate il `create-container-group-definition` comando CLI. Per ulteriori informazioni su questo comando, [create-container-group-definition](#) consultate la AWS CLI Command Reference.

Questo esempio illustra una richiesta per la definizione di un gruppo di contenitori di server di gioco. Si presuppone che tu abbia creato un file JSON con le definizioni dei contenitori per questo gruppo.

```
aws gamelift create-container-group-definition \
  --name MyAdventureGameContainerGroup \
  --operating-system AMAZON_LINUX_2023 \
  --container-group-type GAME_SERVER \
  --total-memory-limit-mebibytes 4096 \
  --total-vcpu-limit 1 \
  --game-server-container-definition file://MyAdventureGameContainers.json
```

Crea un file di definizione del contenitore **JSON**

Quando si crea una definizione di gruppo di contenitori, si definiscono anche i contenitori per il gruppo. Una definizione di contenitore specifica il repository Amazon ECR in cui è archiviata l'immagine del contenitore e le configurazioni opzionali per le porte di rete, i limiti per l'utilizzo di CPU e memoria e altre impostazioni. Consigliamo di creare un singolo JSON file con le configurazioni

per tutti i contenitori in un gruppo di contenitori. La manutenzione di un file è utile per archiviare, condividere e tenere traccia delle versioni di queste configurazioni critiche. Se si utilizza la AWS CLI per creare le definizioni dei gruppi di contenitori, è possibile fare riferimento al file nel comando.

Per creare una definizione di contenitore

1. Crea e apri un nuovo .JSON file. Esempio:

```
[~/work/glc]$ vim SimpleServer.json
```

2. Crea una definizione di contenitore separata per ciascuno dei contenitori del gruppo. Copia il seguente contenuto di esempio e modificalo secondo necessità per i tuoi contenitori. Per i dettagli sulla sintassi di una definizione di contenitore, consulta [ContainerDefinitionInput](#) l'Amazon GameLift Servers API Reference.
3. Salvate il file localmente in modo da potervi fare riferimento in un comando AWS CLI.

Esempio: definizione del contenitore del server di gioco

Example

Questo esempio descrive il contenitore essenziale per il gruppo di contenitori del server di gioco. Il contenitore di replica essenziale include l'applicazione per server di gioco, l'Amazon GameLift Servers Agent, e può includere altro software di supporto per l'hosting dei giochi. La definizione deve includere un nome, un URI dell'immagine e una configurazione di porta. Questo esempio imposta anche alcuni limiti di risorse specifici del contenitore.

```
{
  "ContainerName": "MyAdventureGameServer",
  "ImageUri": "111122223333.dkr.ecr.us-east-1.amazonaws.com/glc-
containers:myadventuregame-server",
  "PortConfiguration": {
    "ContainerPortRanges": [
      {
        "FromPort": 2000,
        "Protocol": "TCP",
        "ToPort": 2010
      }
    ]
  },
  "ServerSdkVersion": "5.2.0"
```


}

Amazon GameLift Servers Flotte ovunque

Usa Anywhere le flotte quando vuoi sfruttare Amazon GameLift Servers le funzionalità delle tue risorse di hosting. Le flotte Anywhere vengono comunemente utilizzate come ambienti di test per lo sviluppo iterativo o insieme alle flotte gestite in una soluzione di hosting ibrida.

Una flotta Anywhere è costituita da un set di risorse di elaborazione (virtuali o fisiche) fornite e gestite dall'utente. I computer possono risiedere in qualsiasi posizione geografica con connettività, da un laptop locale a postazioni remote. Quando si configura una flotta Anywhere, si aggiungono computer al parco computer registrandoli tramite Amazon GameLift Servers. Ogni computer è registrato con il relativo indirizzo IP (o nome DNS) in modo da Amazon GameLift Servers poter stabilire una connessione con esso.

Puoi distribuire il software del server di gioco su una flotta Anywhere installandolo su ogni computer e avviando i processi del server di gioco. Ogni processo del server di gioco avviato stabilisce una connessione al Amazon GameLift Servers servizio e segnala la disponibilità a ospitare una sessione di gioco. È possibile utilizzare gli strumenti di gestione e distribuzione della configurazione esistenti per gestire le attività iniziali di distribuzione e gestione dell'host. Sono necessarie alcune attività aggiuntive da utilizzare con Amazon GameLift Servers, tra cui:

- Registra e annulla la registrazione dei computer per aggiungerli o rimuoverli dal parco dati.
- Conserva i token up-to-date di autenticazione su tutti i computer. I processi del server sul computer li utilizzano per la connessione al servizio. Amazon GameLift Servers

Note

Opzionalmente, implementa la tua flotta Anywhere con l'Amazon GameLift ServersAgent per automatizzare queste attività di gestione chiave. Per informazioni, consulta [Collabora con l'Amazon GameLift Servers agente](#).

Consulta questi argomenti su come configurare e gestire le flotte Anywhere:

- [Roadmap di sviluppo per l'hosting con Anywhere Amazon GameLift Servers](#)
- [Roadmap di sviluppo per l'hosting ibrido con Amazon GameLift Servers](#)

- [Configurazione per lo sviluppo iterativo con Anywhere Amazon GameLift Servers](#)
- [Crea un Amazon GameLift Servers Flotta ovunque](#)
- [Aggiornamento di una configurazione del Amazon GameLift Servers parco veicoli](#)

Workflow di creazione di flotte ovunque

Per le flotte Anywhere, Amazon GameLift Servers imposta solo la risorsa della flotta. L'utente configura e registra i computer con la flotta, installa il software del server di gioco e avvia i processi del server di gioco per ospitare sessioni di gioco.

1. Amazon GameLift Servers crea la risorsa della flotta nella regione di origine della flotta. Lo stato della flotta e lo stato della posizione personalizzata sono impostati su Nuovo.
2. Amazon GameLift Servers inizia a scrivere gli eventi nel registro degli eventi della flotta.
3. Dopo aver creato la risorsa del parco veicoli, Amazon GameLift Servers imposta lo stato della flotta su Attivo. A questo punto, puoi registrare nuovi computer con la flotta.

Crea un Amazon GameLift Servers Flotta ovunque

Questo argomento descrive come creare un Amazon GameLift Servers Flotta ovunque. Con una flotta Anywhere, puoi usare core Amazon GameLift Servers funzionalità di gestione delle sessioni di gioco mentre si ospitano sessioni di gioco con le proprie risorse di elaborazione. Crea una flotta Anywhere per il tuo hardware locale o altre risorse basate sul cloud.

Le flotte Anywhere vengono comunemente utilizzate insieme Amazon GameLift Servers flotte gestite in una soluzione di hosting ibrida. Forniscono inoltre ambienti di test utili per lo sviluppo di un gioco da ospitare con Amazon GameLift Servers. Consulta questi argomenti per saperne di più su quando e come incorporare Amazon GameLift Servers Anywhere flotte in una soluzione di hosting di giochi:

- [Hosting ibrido](#)
- [Implementa flotte di hosting per Amazon GameLift Servers](#)
- [Configurazione per lo sviluppo iterativo con Anywhere Amazon GameLift Servers](#)

Poiché le flotte Anywhere sono autogestite, la creazione di una flotta richiede del lavoro aggiuntivo. Per preparare una flotta Anywhere a ospitare sessioni di gioco e giocatori, devi completare le seguenti attività:

Argomenti

- [Prima di iniziare](#)
- [Creare una posizione personalizzata](#)
- [Crea una flotta Anywhere](#)
- [Aggiungi un computer alla flotta](#)
- [Avvia un server di gioco](#)

Prima di iniziare

Prima di creare una flotta Anywhere, esegui le seguenti operazioni. Per una guida più dettagliata, consulta [Roadmap di sviluppo per l'hosting con Anywhere Amazon GameLift Servers](#) o [Roadmap di sviluppo per l'hosting ibrido con Amazon GameLift Servers](#).

- Integra il codice del tuo server di gioco con il Amazon GameLift Servers server SDK versione 5.x (o superiore). Non è necessario completare tutte le attività di integrazione del gioco, solo quelle necessarie per la creazione di un server di gioco. Una pratica comune consiste nel configurare la macchina locale come una flotta Anywhere e utilizzare un'interfaccia a riga di comando per testare l'integrazione del server di gioco (vedi [Configura i test locali con Amazon GameLift Servers Anywhere](#)). È possibile incorporare componenti aggiuntivi (come Amazon GameLift Servers client di gioco abilitato) man mano che li sviluppi.
- Package del software del server di gioco per l'installazione sui computer della tua flotta Anywhere. Il pacchetto deve includere la versione integrata del server di gioco e tutto il software di supporto necessario per far funzionare il server di gioco.
- Decidi se utilizzare il Amazon GameLift Servers Agente della tua flotta Anywhere. L'Agent è uno strumento di gestione dei processi on-computing che automatizza alcune delle attività chiave relative alla gestione dei processi e dei calcoli del server da utilizzare con Amazon GameLift Servers. Per ulteriori informazioni, vedere [Collabora con l'Amazon GameLift Servers agente](#).

Creare una posizione personalizzata

Crea una posizione personalizzata per rappresentare la posizione fisica delle tue risorse di elaborazione. Quando crei una flotta Anywhere, devi avere almeno una posizione personalizzata già definita. Puoi creare altre sedi personalizzate e aggiungerle a una flotta esistente in qualsiasi momento.

Per creare una sede personalizzata

Usa uno dei Amazon GameLift Servers console o AWS Command Line Interface (AWS CLI) per creare una posizione personalizzata.

Console

Nella [Amazon GameLift Servers console](#), utilizza il riquadro di navigazione per aprire la pagina Posizioni. Scegli Crea posizione per aprire la finestra di dialogo Crea.

1. Nella finestra di dialogo, inserite un nome per la località. Come procedura consigliata, utilizzate un nome che descriva una posizione significativa per un set di risorse di calcolo. Potrebbe trattarsi di posizioni geografiche, il nome di un data center o un altro identificatore di posizione. Amazon GameLift Servers aggiunge il nome della posizione personalizzata con custom -.
2. (Facoltativo) Aggiungi tag alla tua posizione personalizzata. Ogni tag è composto da una chiave e da un valore opzionale, entrambi personalizzabili. Assegna tag alle AWS risorse che desideri classificare in modi utili, ad esempio per scopo, proprietario o ambiente. Scegli Aggiungi nuovo tag per ogni tag che desideri aggiungere.
3. Scegli Create (Crea) .

AWS CLI

Crea una posizione personalizzata utilizzando il [create-location](#) comando. Fornisci un location-name valore, che deve iniziare con custom-. Come procedura consigliata, utilizzate un nome che descriva una posizione significativa per un set di risorse di elaborazione. Potrebbe trattarsi di posizioni geografiche, il nome di un data center o un altro identificatore di posizione.

```
aws gamelift create-location \
  --location-name custom-location-1
```

Output

```
{
  "Location": {
    "LocationName": "custom-location-1",
    "LocationArn": "arn:aws:gamelift:us-east-1:111122223333:location/custom-
location-1"
  }
}
```

Crea una flotta Anywhere

Crea una flotta Anywhere per un set di risorse di elaborazione di tua proprietà. Una nuova flotta Anywhere inizia vuota; aggiungi computer alla flotta registrandoli.

Al momento della creazione, una nuova flotta Anywhere passa rapidamente da uno stato all'altro della flotta. NEW ACTIVE Puoi aggiungere computer alla flotta una volta raggiunta la soglia. ACTIVE

Per creare una flotta Anywhere

Usa uno dei due Amazon GameLift Servers console o AWS Command Line Interface (AWS CLI) per creare una flotta Anywhere.

Console

Nella [Amazon GameLift Servers console](#), usa il pannello di navigazione per aprire la pagina Fleets. Scegli Crea flotta per avviare il flusso di lavoro di creazione della flotta.

Passaggio 1 Scegli il tipo di elaborazione

Seleziona l'opzione Anywhere e scegli Avanti.

Fase 2 Definire i dettagli della flotta

In questo passaggio, specifica alcune impostazioni chiave a livello di flotta.

1. Compila la sezione dei dettagli della flotta:
 - a. Inserisci un nome per la flotta. Ti consigliamo di utilizzare uno schema di denominazione della flotta che faciliti l'identificazione dei tipi di flotta durante la visualizzazione degli elenchi di flotte.
 - b. Fornisci una breve descrizione della flotta.
2. Imposta questi dettagli aggiuntivi opzionali in base alle esigenze. Puoi aggiornare queste impostazioni del parco veicoli in un secondo momento.
 - a. Quando crei una flotta per la produzione o i test preliminari alla produzione, utilizza questa impostazione per specificare un valore di costo orario per i calcoli del parco macchine. Amazon GameLift Servers può utilizzare queste informazioni durante il processo di posizionamento della sessione di gioco per selezionare le risorse di hosting in base ai costi.
 - b. Se desideri combinare i dati metrici per questo parco veicoli e altri, specifica il nome del gruppo Metric. Utilizza lo stesso nome di gruppo metrico per tutte le flotte che

desideri combinare. Visualizza le metriche per il gruppo di metriche per vedere i dati aggregati.

3. Aggiungi tag opzionali alla tua posizione personalizzata. Ogni tag è composto da una chiave e da un valore opzionale, entrambi personalizzabili. Assegna tag alle AWS risorse che desideri classificare in modi utili, ad esempio per scopo, proprietario o ambiente. Scegli Aggiungi nuovo tag per ogni tag che desideri aggiungere.
4. Scegli Avanti per continuare il flusso di lavoro.

Passaggio 3 Seleziona posizioni personalizzate

In questo passaggio, identifica la posizione fisica dei computer che intendi aggiungere a questa flotta. Ora puoi specificare una o più sedi e aggiungere o rimuovere sedi in un secondo momento, se necessario.

1. In Posizioni personalizzate, seleziona una o più sedi per i computer della flotta. L'elenco include tutte le località personalizzate che sono state definite nella selezione Regione AWS corrente. Per definire una nuova posizione personalizzata da aggiungere al parco veicoli, scegli Crea sede.
2. Scegli Avanti per continuare il flusso di lavoro.

Passaggio 4: rivedi e crea

Controlla le impostazioni prima di creare la flotta.

Quando sei pronto per implementare la nuova flotta, scegli Crea. Amazon GameLift Servers avvia immediatamente il processo di attivazione della flotta, assegnando un ID univoco e impostando lo stato della flotta. NEW Puoi monitorare i progressi della flotta nella pagina Flotte.

AWS CLI

Usa il [create-fleet](#) comando per creare una flotta di tipi di calcolo. ANYWHERE Fornisci un nome e almeno una posizione personalizzata. Amazon GameLift Servers crea la risorsa Anywhere fleet con l'impostazione predefinita corrente Regione AWS (oppure puoi aggiungere un tag `--region` per specificarne un altro Regione AWS).

La seguente richiesta di esempio crea una nuova flotta con le impostazioni minime richieste. Sostituisci *FleetName* e *custom-location* con le tue informazioni.

```
aws gamelift create-fleet \  
--name FleetName \  
--compute-type ANYWHERE \  
--locations "Location=custom-location"
```

Example response

```
{  
  "FleetAttributes": {  
    "FleetId": "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",  
    "FleetArn": "arn:aws:gamelift:us-west-2:111122223333:fleet/  
fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",  
    "Name": "HardwareAnywhere",  
    "CreationTime": "2023-02-23T17:57:42.293000+00:00",  
    "Status": "ACTIVE",  
    "MetricGroups": [  
      "default"  
    ],  
    "CertificateConfiguration": {  
      "CertificateType": "DISABLED"  
    },  
    "ComputeType": "ANYWHERE"  
  }  
}
```

Al momento della creazione, una nuova flotta Anywhere passa rapidamente allo status di flottaACTIVE. Puoi aggiungere computer alla flotta dopo che è stata ACTIVE raggiunta.

Nota che la risposta non include le ubicazioni del parco veicoli. Puoi recuperare i dettagli completi della flotta chiamando [describe-fleet-attributes](#) e [describe-fleet-location-attributes](#).

Aggiungi un computer alla flotta

Per aggiungere una risorsa di calcolo a una flotta e prepararla per ospitare sessioni di gioco, esegui le seguenti attività:

- Registrate il computer con il parco macchine. La registrazione dice Amazon GameLift Servers quali risorse fisiche di hosting fanno parte della flotta.

- Richiedi un token di autenticazione per il calcolo. Ogni server di gioco in esecuzione sul computer necessita di questo token per connettersi al Amazon GameLift Servers servizio. I token di autenticazione sono temporanei e devono essere aggiornati regolarmente.

Note

Se stai distribuendo il software del tuo server di gioco con il Amazon GameLift Servers Agente, puoi saltare questo passaggio. L'agente registra automaticamente ogni calcolo e mantiene un token di autenticazione valido per il calcolo. Per informazioni, consulta [Collabora con l'Amazon GameLift Servers agente](#).

Puoi registrare un calcolo e richiedere un token di autenticazione utilizzando la AWS CLI o effettuando chiamate programmatiche all'SDK per AWS Amazon GameLift Servers. Queste azioni non sono disponibili tramite Amazon GameLift Servers console.

Come procedura consigliata, consigliamo di automatizzare entrambe queste attività aggiungendo uno script di avvio a ciascun calcolo. Lo script di avvio chiama automaticamente entrambi i comandi `register-compute` e `get-compute-auth-token`. Puoi anche automatizzare le attività per aggiornare regolarmente il token di autenticazione per tutta la durata del calcolo e annullare la registrazione del calcolo allo spegnimento.

Ciascuna delle azioni di avvio restituisce valori specifici del calcolo che è necessario archiviare sul computer. Quando un processo del server di gioco viene avviato sul computer, deve passare questi valori come parametri del server durante l'inizializzazione di una connessione con Amazon GameLift Servers servizio (vedi [ServerParameters](#) nel riferimento all'SDK del server). Ti consigliamo di impostare questi valori specifici del calcolo (o le relative posizioni memorizzate) come variabili di ambiente. Se stai usando il Amazon GameLift Servers Agente, questo compito è gestito per te. I valori specifici del calcolo sono i seguenti:

- `register-compute` restituisce un valore per `GameLiftServiceSdkEndpoint`. Imposta questo valore sul parametro `websocketUrl` del server.
- `compute-auth-token` restituisce il token di autenticazione. Imposta questo valore sul parametro `authToken` del server.

AWS CLI

Le seguenti istruzioni descrivono come inviare manualmente ogni richiesta utilizzando la AWS CLI.

Per registrare un calcolo

Chiama [register-compute](#) per registrare un computer. Identifica l'ID del parco macchine a cui aggiungere il calcolo. Fornisci le seguenti informazioni di calcolo: un nome, un indirizzo IP e una posizione significativi. La posizione del computer deve essere una posizione personalizzata già associata alla flotta. Se desideri utilizzare una posizione personalizzata diversa, utilizza il Amazon GameLift Servers console per aggiornare la flotta o chiamare il comando AWS CLI [create-fleet-locations](#) per aggiungere una posizione personalizzata alla flotta.

Nell'esempio seguente, sostituisci i valori segnaposto per il tuo calcolo e il tuo parco veicoli. Il `fleet-id` valore viene restituito quando si crea una flotta Anywhere. Puoi recuperare i dettagli completi della flotta chiamando [describe-fleet-attributes](#) e [describe-fleet-location-attributes](#).

```
aws gamelift register-compute \  
  --compute-name HardwareAnywhere \  
  --fleet-id arn:aws:gamelift:us-east-1:111122223333:fleet/  
fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa \  
  --ip-address 10.1.2.3 \  
  --location custom-location-1
```

Output di esempio

```
{  
  "Compute": {  
    "FleetId": "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",  
    "FleetArn": "arn:aws:gamelift:us-west-2:111122223333:fleet/  
fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",  
    "ComputeName": "HardwareAnywhere",  
    "ComputeArn": "arn:aws:gamelift:us-west-2:111122223333:compute/  
HardwareAnywhere",  
    "IpAddress": "10.1.2.3",  
    "ComputeStatus": "Active",  
    "Location": "custom-location-1",  
    "CreationTime": "2023-02-23T18:09:26.727000+00:00",  
    "GameLiftServiceSdkEndpoint": "wss://us-west-2.api.amazongamelift.com"
```

```
}
}
```

Per richiedere un token di autenticazione

Chiama [get-compute-auth-token](#) per richiedere un token di autenticazione valido. Registra un calcolo. Identifica l'ID della flotta e il nome del calcolo.

Nell'esempio seguente, sostituisci i valori segnaposto per il calcolo e la flotta. Il `fleet-id` valore viene restituito quando si crea una flotta Anywhere. Puoi recuperare i dettagli completi della flotta [describe-fleet-attributes](#) chiamando. Per trovare informazioni di calcolo, chiama [list-compute](#) con l'ID della flotta per vedere tutti i computer registrati nel parco macchine.

```
aws gamelift get-compute-auth-token \
  --fleet-id arn:aws:gamelift:us-east-1:111122223333:fleet/
fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa \
  --compute-name HardwareAnywhere
```

Output di esempio

```
{
  "FleetId": "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
  "FleetArn": "arn:aws:gamelift:us-east-1:111122223333:fleet/
fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
  "ComputeName": "HardwareAnywhere",
  "ComputeArn": "arn:aws:gamelift:us-east-1:111122223333:compute/
HardwareAnywhere",
  "AuthToken": "0c728041-3e84-4aaa-b927-a0fb202684c0",
  "ExpirationTimestamp": "2023-02-23T18:47:54+00:00"
}
```

Avvia un server di gioco

Dopo aver creato una flotta Anywhere e aver aggiunto uno o più computer alla flotta, sei pronto per iniziare a utilizzare i tuoi server di gioco.

Passaggio 1 Installa il software del server di gioco

Installa la build del tuo server di gioco e tutto il software dipendente su ogni computer della tua flotta Anywhere. La build del server di gioco deve essere integrata con Amazon GameLift Servers

server SDK versione 5.x (o successiva) con la funzionalità minima richiesta per comunicare con Amazon GameLift Servers servizio.

Passaggio 2 Prepara i tuoi computer per far funzionare un server di gioco

Assicurati che ogni computer sia registrato e disponga di un token di autenticazione valido. Se utilizzi degli script per gestire queste attività, assicurati che gli script vengano eseguiti su ogni computer prima di avviare qualsiasi processo del server di gioco.

Se hai distribuito il Amazon GameLift Servers Agent con il software del server di gioco, assicurati che l'eseguibile Agent venga avviato.

Passaggio 3 Avvia un processo del server di gioco

Esegui un'istanza del file eseguibile del tuo server di gioco su un computer. Se la build del server di gioco è integrata correttamente, il processo del server di gioco richiama l'azione SDK del server `InitSDK()` con una serie di parametri server validi. Quando il processo del server è pronto per ospitare una sessione di gioco, chiama `ProcessReady()`.

Note

Se hai distribuito il software del server di gioco con il Amazon GameLift Servers Agente, puoi saltare questo passaggio. L'agente avvia automaticamente i processi del server di gioco in base alle istruzioni di runtime fornite dall'utente.

È possibile monitorare l'avanzamento visualizzando le metriche dei processi del server per l'attivazione e i processi attivi del server. Per informazioni, consulta [Parametri di Amazon GameLift Servers per parchi istanze](#). Se il processo del server di gioco non riesce a inicializzarsi, verifica che il processo stia recuperando i valori dei parametri del server corretti per il computer su cui è in esecuzione.

Crea una soluzione di hosting ibrida

Una soluzione di hosting ibrido combina diverse fonti di risorse di hosting di giochi per ospitare i tuoi giochi, tra cui flotte Amazon GameLift Servers gestite che girano in rete Cloud AWS e risorse che fornisci e gestisci autonomamente. Questo argomento descrive alcuni modelli comuni per la creazione di una soluzione ibrida e fornisce suggerimenti su come combinare con successo l'hosting di giochi autogestito con l'hosting di giochi basato su cloud gestito da Amazon GameLift Servers

Schemi ibridi comuni

- Ottimizzazione dei costi: utilizza il tipo di flotta più conveniente per ogni scenario, ad esempio flotte Anywhere per la capacità di base e flotte gestite per i picchi di domanda.
- Flessibilità geografica: implementa flotte gestite in flotte ad alto traffico Regioni AWS e Anywhere in cui sono presenti infrastrutture esistenti o requisiti di conformità specifici.
- Mitigazione del rischio: riduci la dipendenza da un singolo approccio di hosting distribuendo il carico su più tipi di flotta e fornitori.
- Migrazione graduale: transizione da soluzioni di hosting locali o di altro tipo a soluzioni di hosting AWS graduale, mantenendo la continuità del servizio.

Suddivisione tra sviluppo e produzione

Utilizza diversi tipi di flotta per gli ambienti di sviluppo e produzione:

- Sviluppo: flotte ovunque per uno sviluppo e un collaudo convenienti
- Produzione: Amazon gestito EC2 o flotte di container per un hosting di produzione scalabile e affidabile

Ottimizzazione regionale

Ottimizza i tipi di flotta in base Regioni AWS alle caratteristiche:

- Traffico elevato Regioni AWS: flotte gestite con auto-scaling per una domanda variabile
- Specializzato Regioni AWS: flotte ovunque per garantire la conformità, la sovranità dei dati o l'infrastruttura esistente

Suddivisione della capacità su più livelli

Utilizza diversi tipi di flotta per diversi livelli di capacità:

- Capacità di base: flotte o istanze riservate ovunque per un carico prevedibile
- Capacità burst: flotte gestite con auto-scaling per i picchi di domanda
- Capacità di overflow: istanze spot o aggiuntive per picchi estremi Regioni AWS

Considerazioni sull'implementazione

Quando crei una soluzione ibrida, considera questi fattori chiave:

Configurazione della coda delle sessioni di gioco

Configura le code delle sessioni di gioco per includere tutti i tipi di flotta e imposta le priorità e le preferenze di latenza appropriate per garantire un posizionamento ottimale nell'infrastruttura ibrida.

Monitoraggio e osservabilità

Implementa un monitoraggio completo su tutti i tipi di flotta per mantenere la visibilità su prestazioni, capacità e costi della tua soluzione ibrida.

Complessità operativa

Tenete conto della maggiore complessità operativa derivante dalla gestione di più tipi di flotta, inclusi diversi processi di implementazione, strumenti di monitoraggio e procedure di risoluzione dei problemi.

La connettività di rete

Garantisce una connettività di rete affidabile tra i diversi ambienti di hosting, in particolare per le flotte Anywhere che possono essere locali o in diversi provider di cloud.

Guida introduttiva all'hosting ibrido

Per implementare una soluzione di hosting ibrido:

1. Inizia in modo semplice: inizia con un solo tipo di parco veicoli e aggiungine altri gradualmente man mano che le tue esigenze diventano più chiare.
2. Pianifica la tua architettura: progetta la tua architettura ibrida in base ai tuoi requisiti specifici di costi, prestazioni, conformità e complessità operativa.
3. Configura le code: imposta code per le sessioni di gioco che coprono i diversi tipi di flotta con priorità e strategie di posizionamento appropriate.
4. Esegui test approfonditi: verifica il posizionamento delle sessioni di gioco e gli scenari di failover sulla tua infrastruttura ibrida prima di passare alla produzione.
5. Monitoraggio e ottimizzazione: monitora continuamente le prestazioni e i costi di tutti i tipi di flotta e regola la configurazione secondo necessità.

Suggerimenti

- Utilizza gli stessi componenti client e server di gioco con risorse di hosting gestite e autogestite. Offri ai giocatori un'esperienza unificata su tutte le risorse di hosting.
- Usa gli stessi FlexMatch matchmaker per piazzare le partite su tutte le risorse di hosting.
- Gestisci centralmente le tue risorse di hosting ibrido insieme mentre le distribuisce in tutto il mondo.
- Man mano che la domanda dei giocatori oscilla, gestisci senza problemi i carichi delle sessioni di gioco tra risorse gestite e autogestite.
- Con l'Amazon GameLift ServersAgent, puoi utilizzare gli stessi strumenti per gestire i cicli di vita dei server di gioco su tutti i tipi di risorse di hosting.
- Raccogli le metriche e i registri di gioco e giocatori su tutte le risorse di hosting. Sfrutta Amazon GameLift Servers le funzionalità e gli altri AWS servizi per combinare i dati e sviluppare soluzioni di osservabilità coerenti.

Configura il posizionamento delle sessioni di gioco

Il posizionamento delle sessioni di gioco è il processo di ricerca dei server di gioco disponibili per ospitare nuove sessioni di gioco. Amazon GameLift Servers utilizza le code delle sessioni di gioco per posizionare in modo intelligente le sessioni di gioco tra le flotte, tenendo conto di fattori come la latenza dei giocatori, la capacità della flotta e l'ottimizzazione dei costi.

Una coda di sessioni di gioco ben configurata assicura che i giocatori siano connessi al miglior server di gioco disponibile, offrendo prestazioni ottimali e facendo un uso efficiente delle risorse di hosting. Le code possono estendersi su più flotte e AWS regioni per fornire copertura e ridondanza globali.

La coda delle sessioni di gioco è il meccanismo principale Amazon GameLift Servers utilizzato per cercare i server di gioco disponibili e sceglierli per ospitare nuove sessioni di gioco. Queues offre un modo molto più efficiente per elaborare un gran numero di richieste di sessioni di gioco e trovare posizioni per esse su più flotte di risorse di hosting. Se la tua soluzione di hosting utilizza più di una flotta e stai elaborando volumi elevati di richieste, probabilmente hai bisogno di una coda.

Quando il gioco desidera iniziare una nuova sessione di gioco per i giocatori, invia una richiesta di posizionamento al Amazon GameLift Servers servizio, che la invia alla coda. La configurazione della coda determina quando e come vengono elaborate le richieste. Durante l'elaborazione di una richiesta di posizionamento, Amazon GameLift Servers cerca in un insieme di flotte un server di gioco per ospitare la sessione di gioco. Il posizionamento ha esito positivo quando Amazon GameLift Servers trova un server di gioco disponibile e gli chiede di avviare una sessione di gioco.

Argomenti

- [Caratteristiche della coda](#)
- [Procedure consigliate per le code delle sessioni di Amazon GameLift Servers gioco](#)
- [Crea una coda per le sessioni di gioco](#)
- [Imposta la notifica degli eventi per il posizionamento della sessione di gioco](#)

Caratteristiche della coda

Una coda di sessione di Amazon GameLift Servers gioco è una risorsa AWS cloud. Puoi creare una coda in qualsiasi Amazon GameLift Servers formato Regione AWS che supporti (vedi [Amazon GameLift Servers sedi di assistenza](#)). Le richieste di posizionamento delle sessioni di gioco vengono inviate a quella posizione ed elaborate lì.

L'automazione del posizionamento delle sessioni di gioco con le code offre vantaggi significativi sia per gli sviluppatori di giochi che per i giocatori. Ciò include:

- Le code offrono il «miglior posizionamento possibile». Durante l'elaborazione delle richieste di posizionamento delle sessioni di gioco, una coda utilizza l'algoritmo Amazon GameLift Servers FleetIQ per dare priorità ai posizionamenti in base a una serie di preferenze definite, tra cui costo, posizione e latenza del giocatore.
- Le code supportano le flotte Spot per contribuire a ridurre i costi di hosting dei giochi. Puoi configurare le code con le flotte AWS Spot, che spesso offrono costi di hosting notevolmente inferiori, e con le flotte On-Demand. Poiché il basso costo è uno dei criteri chiave per i posizionamenti, le code possono sempre sfruttare le differenze di costo.
- Le code possono inserire nuovi giochi più velocemente in caso di forte richiesta. Configurando una coda con più flotte, offri opzioni più flessibili per il posizionamento delle sessioni di gioco. Ma le flotte aggiuntive forniscono anche la capacità di backup necessaria quando la domanda aumenta. Per qualsiasi richiesta di collocamento, se non è Amazon GameLift Servers possibile collocare una sessione di gioco nella posizione preferita, passa automaticamente alla valutazione delle altre sedi.
- Le code possono rendere più resistente la disponibilità dei server di gioco. Le interruzioni possono verificarsi. Con una coda composta da più flotte, un rallentamento o un'interruzione non devono influire sull'accesso dei giocatori al gioco. Configurando la coda con flotte capienti in diverse Regioni AWS zone di disponibilità, puoi fare in modo che i giocatori possano sempre trovare una sessione di gioco a cui partecipare.

- Ottieni metriche sui posizionamenti delle sessioni di gioco e sulle prestazioni in coda. Amazon GameLift Serversemette metriche sulla coda, tra cui statistiche sui piazzamenti riusciti e non riusciti, il numero di richieste in coda e il tempo medio che le richieste trascorrono in coda. È possibile visualizzare questi parametri nella console Amazon GameLift Servers o in CloudWatch.

Per iniziare a creare una coda iniziale di base, consulta. [Crea una coda per le sessioni di gioco](#)

Argomenti

- [Procedure consigliate per le code delle sessioni di Amazon GameLift Servers gioco](#)
- [Crea una coda per le sessioni di gioco](#)
- [Imposta la notifica degli eventi per il posizionamento della sessione di gioco](#)

Procedure consigliate per le code delle sessioni di Amazon GameLift Servers gioco

Una coda per le sessioni di gioco contiene un elenco di flotte in cui è Amazon GameLift Servers possibile inserire nuove sessioni di gioco. Ogni flotta può avere risorse di hosting distribuite in più località geografiche. Quando si sceglie una posizione, la coda seleziona una flotta e un'ubicazione della flotta in base a una serie di priorità impostate per la flotta.

Prendi in considerazione le seguenti linee guida e best practice:

- Aggiungi flotte in località adatte ai tuoi giocatori. Puoi aggiungere flotte e alias in qualsiasi località disponibile. La posizione è importante se effettui posizionamenti in base alla latenza riportata dai giocatori.
- Usa gli alias per tutte le flotte. Assegna un alias a ogni flotta in coda e usa i nomi degli alias per impostare le destinazioni nella coda.
- Usa una build o uno script di gioco uguali o simili per tutte le flotte. La coda potrebbe mettere i giocatori in sessioni di gioco su qualsiasi flotta in coda. I giocatori devono essere in grado di giocare in qualsiasi sessione di gioco su qualsiasi flotta.
- Crea flotte in almeno due località. Avendo server di gioco ospitati in almeno un'altra località, riduci l'impatto delle interruzioni regionali sui tuoi giocatori. È possibile ridurre le dimensioni delle flotte di backup e utilizzare la scalabilità automatica per aumentare la capacità in caso di aumento dell'utilizzo.

- Dai priorità al posizionamento della sessione di gioco. Una coda dà priorità alle scelte di posizionamento in base a diversi elementi, incluso l'ordine dell'elenco delle destinazioni.
- Crea la tua coda nella stessa posizione del servizio clienti. Mettendo la coda in una posizione vicina al servizio clienti, puoi ridurre al minimo la latenza di comunicazione.
- Utilizza flotte con più sedi. Usa la configurazione del filtro di coda per evitare che la coda collochi le sessioni di gioco in posizioni specifiche. Puoi utilizzare almeno due flotte con più sedi con sedi diverse per mitigare l'impatto dei piazzamenti di gioco durante un'interruzione regionale.
- Utilizza la stessa impostazione del certificato TLS per tutte le flotte. I client di gioco che si connettono alle sessioni di gioco delle tue flotte devono disporre di protocolli di comunicazione compatibili.

Crea una coda per le sessioni di gioco

Le code vengono utilizzate per effettuare nuove sessioni di gioco su più flotte e località. Il gioco avvia nuove sessioni di gioco inviando richieste di posizionamento a una coda. Una coda è configurata con istruzioni su come elaborare le richieste. Scopri di più sull'avvio delle richieste di posizionamento per le sessioni di gioco in [Crea sessioni di gioco](#)

Per creare una coda per le sessioni di gioco

Queste istruzioni illustrano come creare una coda di lavoro semplice con impostazioni di configurazione minime e impostazioni predefinite. Esistono diverse opzioni per personalizzare la configurazione di una coda. Queste opzioni ti aiutano a ottenere i migliori posizionamenti possibili in base alle esigenze del tuo gioco. Per ulteriori informazioni sulla personalizzazione delle code per il gioco, consulta [Personalizza una coda di sessioni di gioco](#). Puoi aggiornare la maggior parte delle impostazioni di configurazione delle code in qualsiasi momento.

Puoi creare una coda di sessioni di gioco utilizzando la Amazon GameLift Servers console o la AWS CLI.

Console

Nella [Amazon GameLift Serversconsole](#), seleziona una AWS regione in cui lavorare. Apri la barra di navigazione a sinistra della console e scegli Code.

1. Nella pagina Code, scegli Crea coda per avviare il flusso di lavoro.
2. In Impostazioni coda, inserisci le seguenti impostazioni:

- a. Immettete un nome per la coda. Questo nome deve essere univoco rispetto a Regione AWS quello in cui stai creando la coda.
 - b. Mantieni l'impostazione di timeout predefinita, che è 600 secondi (o 10 minuti). Questo valore controlla per quanto tempo si Amazon GameLift Servers cerca di effettuare una nuova sessione di gioco prima di interromperla. Amazon GameLift Servers cerca le risorse disponibili fino al timeout della richiesta. È possibile aggiornare l'impostazione del timeout di una coda in qualsiasi momento.
 - c. Salta la sezione relativa ai criteri di latenza di Player. Una coda utilizza i criteri di latenza solo quando riceve richieste di posizionamento che includono dati sulla latenza dei giocatori. Puoi aggiungere politiche di latenza a una coda in qualsiasi momento. Per ulteriori informazioni sulla creazione di criteri di latenza, consulta. [Crea una politica di latenza dei giocatori](#)
3. Salta la sezione Posizionamento delle sessioni di gioco per utilizzare l'impostazione predefinita di Tutte le posizioni. Questa impostazione consente di creare un elenco di posizioni consentite in cui la coda può effettuare posizionamenti (chiamata anche configurazione dei filtri). Per ulteriori informazioni sulla definizione delle priorità in base alla posizione e alle configurazioni dei filtri, consulta. [Assegna priorità ai posizionamenti in base alla posizione](#)
 4. In Ordine di destinazione, aggiungi una o più flotte alla coda. Puoi identificare le flotte utilizzando fleet IDs o ARNs, o utilizzando un alias di flotta. Quando aggiungi più flotte, tieni presente che tutte dovrebbero eseguire build di gioco simili ed essere compatibili con qualsiasi client di gioco che utilizza questa coda. Inoltre, tutte le flotte in coda devono avere la stessa configurazione di certificati.
 - a. Seleziona la regione in cui è stata creata la flotta o l'alias. Per una flotta con più sedi, questa è la regione «di origine».
 - b. Per Tipo di destinazione, seleziona una flotta o un alias.
 - c. Le selezioni relative alla regione e al tipo compongono un elenco a discesa di flotte o alias esistenti. Selezionane uno da designare come destinazione della coda.
 - d. Per specificare un'altra flotta o un alias per la coda, scegli Aggiungi destinazione e ripeti i passaggi precedenti.
 - e. Dopo aver aggiunto un elenco di destinazioni, utilizza la drag-and-drop funzione per riordinare le destinazioni. Amazon GameLift Servers utilizza questo ordine per assegnare priorità ai posizionamenti in base alla destinazione.

5. Salta la sezione Priorità di posizionamento della sessione di gioco per mantenere l'ordine di priorità predefinito. Questa impostazione ti consente di personalizzare il modo in cui Amazon GameLift Servers scegli dove cercare le risorse di hosting disponibili per i nuovi posizionamenti delle sessioni di gioco. Per ulteriori informazioni sull'assegnazione di priorità ai posizionamenti, consulta [Dare la priorità al posizionamento delle sessioni di gioco](#). Puoi aggiornare le priorità di posizionamento di una coda in qualsiasi momento.
6. In Ordine di ubicazione, mantieni i valori predefiniti. Questa impostazione viene utilizzata per stabilire le priorità in base alla posizione del parco veicoli. Fornisce l'ordine di posizione da utilizzare. Quando si utilizzano le impostazioni di priorità predefinite, la posizione viene utilizzata come punto di partenza quando la destinazione preferita è una flotta con più sedi.
7. Salta la sezione opzionale delle impostazioni di notifica degli eventi. Le notifiche degli eventi sono necessarie per le code che elaborano un volume elevato di richieste di posizionamento. Per le code che elaborano volumi ridotti, ad esempio per scopi di sviluppo o test, puoi monitorare lo stato delle richieste di collocamento effettuando un sondaggio con [DescribeGameSessionPlacement](#). Per ulteriori dettagli, consultare [Imposta la notifica degli eventi per il posizionamento della sessione di gioco](#). Puoi aggiornare le impostazioni di notifica degli eventi di una coda in qualsiasi momento.
8. Scegli Crea per generare una nuova coda con una personalizzazione minima.

AWS CLI

Example Crea una coda

L'esempio seguente crea una coda di sessioni di gioco con queste configurazioni:

- Un timeout di cinque minuti.
- Due destinazioni per la flotta.
- Filtra per consentire solo i posizionamenti in queste località: `us-east-1`, `us-east-2`, `us-west-2`, e `ca-central-1`.
- Ordine di priorità basato sul costo e quindi sulle ubicazioni in un ordine specificato.

```
aws gamelift create-game-session-queue \  
  --name "sample-test-queue" \  
  --timeout-in-seconds 300 \  
  --placement-strategy "PRIORITY" \  
  --placement-strategy-parameters "us-east-1,us-east-2,us-west-2,ca-central-1" \  
  --placement-strategy-parameters-order "1,2,3,4" \  
  --placement-strategy-parameters-weight "1,2,3,4" \  
  --placement-strategy-parameters-cost "1,2,3,4" \  
  --placement-strategy-parameters-availability "1,2,3,4" \  
  --placement-strategy-parameters-performance "1,2,3,4" \  
  --placement-strategy-parameters-latency "1,2,3,4" \  
  --placement-strategy-parameters-availability-zone "us-east-1a,us-east-1b,us-east-1c,us-east-1d,us-east-1e,us-east-1f,us-east-1g,us-east-1h,us-east-1i,us-east-1j,us-east-1k,us-east-1l,us-east-1m,us-east-1n,us-east-1o,us-east-1p,us-east-1q,us-east-1r,us-east-1s,us-east-1t,us-east-1u,us-east-1v,us-east-1w,us-east-1x,us-east-1y,us-east-1z,us-east-2a,us-east-2b,us-east-2c,us-east-2d,us-east-2e,us-east-2f,us-east-2g,us-east-2h,us-east-2i,us-east-2j,us-east-2k,us-east-2l,us-east-2m,us-east-2n,us-east-2o,us-east-2p,us-east-2q,us-east-2r,us-east-2s,us-east-2t,us-east-2u,us-east-2v,us-east-2w,us-east-2x,us-east-2y,us-east-2z,us-west-2a,us-west-2b,us-west-2c,us-west-2d,us-west-2e,us-west-2f,us-west-2g,us-west-2h,us-west-2i,us-west-2j,us-west-2k,us-west-2l,us-west-2m,us-west-2n,us-west-2o,us-west-2p,us-west-2q,us-west-2r,us-west-2s,us-west-2t,us-west-2u,us-west-2v,us-west-2w,us-west-2x,us-west-2y,us-west-2z,ca-central-1a,ca-central-1b,ca-central-1c,ca-central-1d,ca-central-1e,ca-central-1f,ca-central-1g,ca-central-1h,ca-central-1i,ca-central-1j,ca-central-1k,ca-central-1l,ca-central-1m,ca-central-1n,ca-central-1o,ca-central-1p,ca-central-1q,ca-central-1r,ca-central-1s,ca-central-1t,ca-central-1u,ca-central-1v,ca-central-1w,ca-central-1x,ca-central-1y,ca-central-1z"
```

```
--destinations DestinationArn="arn:aws:gamelift:us-east-1:111122223333:fleet/
fleet-772266ba-8c82-4a6e-b620-a74a62a93ff8" DestinationArn="arn:aws:gamelift:us-
east-1:111122223333:fleet/fleet-33f28fb6-aa8b-4867-85b4-ceb217bf5994" \
--filter-configuration "AllowedLocations=us-east-1, ca-central-1, us-east-2, us-
west-2" \
--priority-configuration PriorityOrder="COST","LOCATION",LocationOrder="us-
east-1","us-east-2","ca-central-1","us-west-2" \
--notification-target "arn:aws:sns:us-east-1:111122223333:gamelift-test.fifo"
```

Note

Puoi ottenere i valori ARN di fleet e alias chiamando uno dei [describe-fleet-attributes](#) o [describe-alias](#) con l'ID fleet o alias.

Se la `create-game-session-queue` richiesta ha esito positivo, Amazon GameLift Servers restituisce un [GameSessionQueue](#) oggetto con la nuova configurazione della coda. È ora possibile inviare richieste alla coda utilizzando [StartGameSessionPlacement](#).

Example Crea una coda con le politiche di latenza dei giocatori

L'esempio seguente crea una coda di sessioni di gioco con queste configurazioni:

- Un timeout di dieci minuti
- Tre destinazioni per la flotta
- Una serie di politiche di latenza dei giocatori

```
aws gamelift create-game-session-queue \
  --name "matchmaker-queue" \
  --timeout-in-seconds 600 \
  --destinations DestinationArn=arn:aws:gamelift:us-east-1::alias/alias-a1234567-
b8c9-0d1e-2fa3-b45c6d7e8910 \
    DestinationArn=arn:aws:gamelift:us-west-2::alias/alias-b0234567-
c8d9-0e1f-2ab3-c45d6e7f8901 \
    DestinationArn=arn:aws:gamelift:us-west-2::fleet/fleet-f1234567-
b8c9-0d1e-2fa3-b45c6d7e8912 \
  --player-latency-policies
  "MaximumIndividualPlayerLatencyMilliseconds=50,PolicyDurationSeconds=120" \
  "MaximumIndividualPlayerLatencyMilliseconds=100,PolicyDurationSeconds=120" \
```

```
"MaximumIndividualPlayerLatencyMilliseconds=150" \
```

Se la `create-game-session-queue` richiesta ha esito positivo, Amazon GameLift Servers restituisce un [GameSessionQueue](#) oggetto con la nuova configurazione della coda.

Imposta la notifica degli eventi per il posizionamento della sessione di gioco

Puoi utilizzare le notifiche degli eventi per monitorare lo stato delle singole richieste di piazzamento. Ti consigliamo di impostare le notifiche relative agli eventi per tutti i giochi con un elevato volume di attività di posizionamento.

Esistono due opzioni per configurare le notifiche degli eventi.

- Fai Amazon GameLift Servers pubblicare notifiche di eventi su un argomento di Amazon Simple Notification Service (Amazon SNS) utilizzando una coda.
- Usa EventBridge gli eventi Amazon pubblicati automaticamente e la sua suite di strumenti per la gestione degli eventi.

Per un elenco degli eventi di posizionamento delle sessioni di gioco emessi da Amazon GameLift Servers, consulta [Eventi di collocamento delle sessioni di gioco](#).

Important

Per i sistemi di posizionamento ad alto volume, consigliamo di utilizzare argomenti Amazon SNS standard (non FIFO) anziché argomenti FIFO. Gli argomenti FIFO hanno limiti di pubblicazione inferiori rispetto agli argomenti standard, il che può comportare eccezioni limitate in caso di carico elevato. Se riscontri problemi di limitazione con gli argomenti FIFO, potresti perdere le notifiche di posizionamento in coda.

Configura un argomento SNS

Amazon GameLift Servers Per pubblicare tutti gli eventi generati da una coda di sessione di gioco su un argomento, imposta il campo di destinazione della notifica su un argomento.

Per configurare un argomento SNS per la notifica di eventi Amazon GameLift Servers

1. [Accedi a Console di gestione AWS e apri la console Amazon SNS nella versione v3/home. https://console.aws.amazon.com/sns/](https://console.aws.amazon.com/sns/)
2. Dalla pagina Argomenti SNS, scegli Crea argomento e segui le istruzioni per creare il tuo argomento.
3. In Criteri di accesso, procedi come segue:
 - a. Scegli il metodo Avanzato.
 - b. Aggiungi la seguente sezione in grassetto dell'oggetto JSON alla policy esistente.

JSON

```
{
  "Version": "2012-10-17",
  "Id": "__default_policy_ID",
  "Statement": [
    {
      "Sid": "__default_statement_ID",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": [
        "SNS:GetTopicAttributes",
        "SNS:SetTopicAttributes",
        "SNS:AddPermission",
        "SNS:RemovePermission",
        "SNS:DeleteTopic",
        "SNS:Subscribe",
        "SNS:ListSubscriptionsByTopic",
        "SNS:Publish"
      ],
      "Resource": "arn:aws:sns:us-east-1:111122223333:your_topic_name",
      "Condition": {
        "StringEquals": {
          "AWS:SourceAccount": "your_account"
        }
      }
    }
  ],
  {
```

```

    "Sid": "__console_pub_0",
    "Effect": "Allow",
    "Principal": {
        "Service": "gamelift.amazonaws.com"
    },
    "Action": "sns:Publish",
    "Resource": "arn:aws:sns:us-east-1:111122223333:your_topic_name",
    "Condition": {
        "ArnLike": {
            "aws:SourceArn": "arn:aws:gamelift:us-east-1:111122223333:gamesessionqueue/your_queue_name"
        }
    }
}
]
}

```

- c. (Facoltativo) Aggiungete un ulteriore controllo degli accessi all'argomento aggiungendo condizioni alla politica delle risorse.
4. Scegli Create topic (Crea argomento).
5. Dopo aver creato l'argomento SNS, aggiungilo alle code durante la creazione della coda oppure modifica una coda esistente per aggiungerlo.

Configura un argomento SNS con crittografia lato server

La crittografia lato server (SSE) consente di archiviare dati sensibili in argomenti crittografati. SSE protegge il contenuto dei messaggi negli argomenti Amazon SNS utilizzando chiavi gestite in AWS Key Management Service (AWS KMS). Per ulteriori informazioni sulla crittografia lato server con Amazon SNS, [consulta Encryption at rest](#) nella Amazon Simple Notification Service Developer Guide.

Per impostare un argomento SNS con crittografia lato server, consulta i seguenti argomenti:

- [Creazione di una chiave nella Guida per](#) gli sviluppatori AWS Key Management Service
- [Abilitazione di SSE per un argomento](#) nella Amazon Simple Notification Service Developer Guide

Quando crei la tua chiave KMS, utilizza la seguente politica per le chiavi KMS:

```

{
    "Effect": "Allow",
    "Principal": {

```

```

    "Service": "gamelift.amazonaws.com"
  },
  "Action": [
    "kms:Decrypt",
    "kms:GenerateDataKey"
  ],
  "Resource": "*",
  "Condition": {
    "ArnLike": {
      "aws:SourceArn":
"arn:aws:gamelift:your_region:your_account:gamesessionqueue/your_queue_name"
    },
    "StringEquals": {
      "kms:EncryptionContext:aws:sns:topicArn":
"arn:aws:sns:your_region:your_account:your_sns_topic_name"
    }
  }
}

```

Configura EventBridge

Amazon GameLift Servers pubblica automaticamente tutti gli eventi di posizionamento delle sessioni di gioco su EventBridge. Con EventBridge puoi impostare regole per indirizzare gli eventi alle destinazioni per l'elaborazione. Ad esempio, puoi impostare una regola per indirizzare l'evento `PlacementFulfilled` a una AWS Lambda funzione che gestisce le attività che precedono la connessione a una sessione di gioco. Per ulteriori informazioni su EventBridge, consulta [What is Amazon EventBridge?](#) nella Amazon EventBridge User Guide.

Di seguito sono riportati alcuni esempi di EventBridge regole da utilizzare con le Amazon GameLift Servers code:

Corrisponde agli eventi di tutte le code Amazon GameLift Servers

```

{
  "source": [
    "aws.gamelift"
  ],
  "detail-type": [
    "GameLift Queue Placement Event"
  ]
}

```


Corrisponde agli eventi di una coda specifica

```
{
  "source": [
    "aws.gamelift"
  ],
  "detail-type": [
    "GameLift Queue Placement Event"
  ],
  "resources": [
    "arn:aws:gamelift:your_region:your_account:gamesessionqueue/your_queue_name"
  ]
}
```

Implementazione di riferimento con AWS CDK

Per un'implementazione di riferimento completa del posizionamento delle sessioni di gioco basate su eventi utilizzando Amazon SNS e Amazon DynamoDB AWS Lambda, consulta la guida al posizionamento delle [sessioni di gioco basate su eventi](#) in Amazon Toolkit. GameLift Questa guida include un AWS CDK modello per l'implementazione di un sistema completo di posizionamento delle sessioni di gioco basato sugli eventi con una AWS Lambda funzione per l'elaborazione degli eventi di posizionamento delle sessioni di gioco, una tabella Amazon DynamoDB per il monitoraggio dello stato di posizionamento delle sessioni di gioco e le migliori pratiche per la configurazione delle code. Amazon GameLift Servers

Personalizzalo in base alla tua soluzione di hosting di giochi

Con una soluzione di hosting di giochi di base, utilizza i seguenti argomenti per personalizzarla e migliorarla per migliorare l'esperienza del giocatore, ottimizzare i costi e aggiungere funzionalità avanzate. Questa sezione descrive varie opzioni di personalizzazione organizzate in base al componente su cui influiscono principalmente. Scegli le personalizzazioni più adatte ai requisiti di gioco e alla base di giocatori.

Argomenti

- [Personalizzazioni della build del server di gioco](#)
 - [Connect il server di gioco Amazon GameLift Servers ospitato ad altre AWS risorse](#)
 - [Consenti al tuo server di gioco di accedere ai dati Amazon GameLift Servers della flotta](#)
 - [Configurare il peering VPC per Amazon GameLift Servers](#)

- [Sessioni dei giocatori e personalizzazioni del matchmaking](#)
 - [Genera giocatore IDs](#)
 - [Aggiungi FlexMatch matchmaking a Amazon GameLift Servers](#)
- [Personalizzazioni del posizionamento delle sessioni di gioco](#)
 - [Personalizza una coda di sessioni di gioco](#)
 - [Dare la priorità al posizionamento delle sessioni di gioco](#)
 - [Crea una coda per le istanze Spot](#)
- [Personalizzazioni delle risorse di hosting](#)
 - [Scegli le risorse di elaborazione per una flotta gestita](#)
 - [Personalizza una flotta di Amazon GameLift Servers container](#)
 - [Riduci i costi di hosting dei giochi con le flotte Spot](#)
 - [Ottimizza la configurazione del runtime del server di gioco su server gestiti Amazon GameLift Servers](#)
 - [Collabora con l'Amazon GameLift Servers agente](#)
 - [Designazione astratta e Amazon GameLift Servers della flotta con un alias](#)

Personalizzazioni della build del server di gioco

Le personalizzazioni della build dei server di gioco offrono l'opportunità di migliorare le funzionalità del server di gioco, incluso l'utilizzo di altri AWS servizi. Queste personalizzazioni estendono le funzionalità del server di gioco oltre l'hosting di base per supportare funzionalità e integrazioni avanzate.

Connect il server di gioco Amazon GameLift Servers ospitato ad altre AWS risorse

Quando crei una build di server di gioco da distribuire su Amazon GameLift Servers flotte, potresti volere che le applicazioni della build di gioco comunichino direttamente e in modo sicuro con altre AWS risorse di tua proprietà. Poiché Amazon GameLift Servers gestisce le tue flotte di giochi che ospitano giochi, devi concedere un accesso Amazon GameLift Servers limitato a queste risorse e servizi.

Alcuni scenari di esempio includono:

- Utilizza un CloudWatch agente Amazon per raccogliere metriche, log e tracce dalle flotte EC2 e dalle flotte Anywhere gestite.

- Invia i dati di log dell'istanza ad Amazon CloudWatch Logs.
- Ottieni file di gioco archiviati in un bucket Amazon Simple Storage Service (Amazon S3).
- Leggi e scrivi dati di gioco (come modalità di gioco o inventario) archiviati in un database Amazon DynamoDB o in un altro servizio di archiviazione dati.
- Invia segnali direttamente a un'istanza utilizzando Amazon Simple Queue Service (Amazon SQS).
- Accedi a risorse personalizzate distribuite e in esecuzione su Amazon Elastic Compute Cloud (Amazon EC2).

Amazon GameLift Servers supporta questi metodi per stabilire l'accesso:

- [Accedi alle AWS risorse con un ruolo IAM](#)
- [Accedi alle AWS risorse con il peering VPC](#)

Accedi alle AWS risorse con un ruolo IAM

Utilizza un ruolo IAM per specificare chi può accedere alle tue risorse e impostare limiti a tale accesso. Le parti fidate possono «assumere» un ruolo e ottenere credenziali di sicurezza temporanee che le autorizzano a interagire con le risorse. Quando le parti effettuano richieste API relative alla risorsa, devono includere le credenziali.

Per configurare l'accesso controllato da un ruolo IAM, esegui le seguenti attività:

1. [Crea il ruolo IAM](#)
2. [Modifica le applicazioni per acquisire credenziali](#)
3. [Associa una flotta al ruolo IAM](#)

Crea il ruolo IAM

In questo passaggio, crei un ruolo IAM, con una serie di autorizzazioni per controllare l'accesso alle AWS risorse e una politica di fiducia che concede Amazon GameLift Servers i diritti di utilizzo delle autorizzazioni del ruolo.

Per istruzioni su come configurare il ruolo IAM, consulta. [Configurare un ruolo di servizio IAM per Amazon GameLift Servers](#) Quando crei la politica delle autorizzazioni, scegli i servizi, le risorse e le azioni specifici con cui le tue applicazioni devono lavorare. Come procedura ottimale, limita il più possibile l'ambito delle autorizzazioni.

Dopo aver creato il ruolo, prendi nota dell'Amazon Resource Name (ARN) del ruolo. È necessario il ruolo ARN durante la creazione della flotta.

Modifica le applicazioni per acquisire credenziali

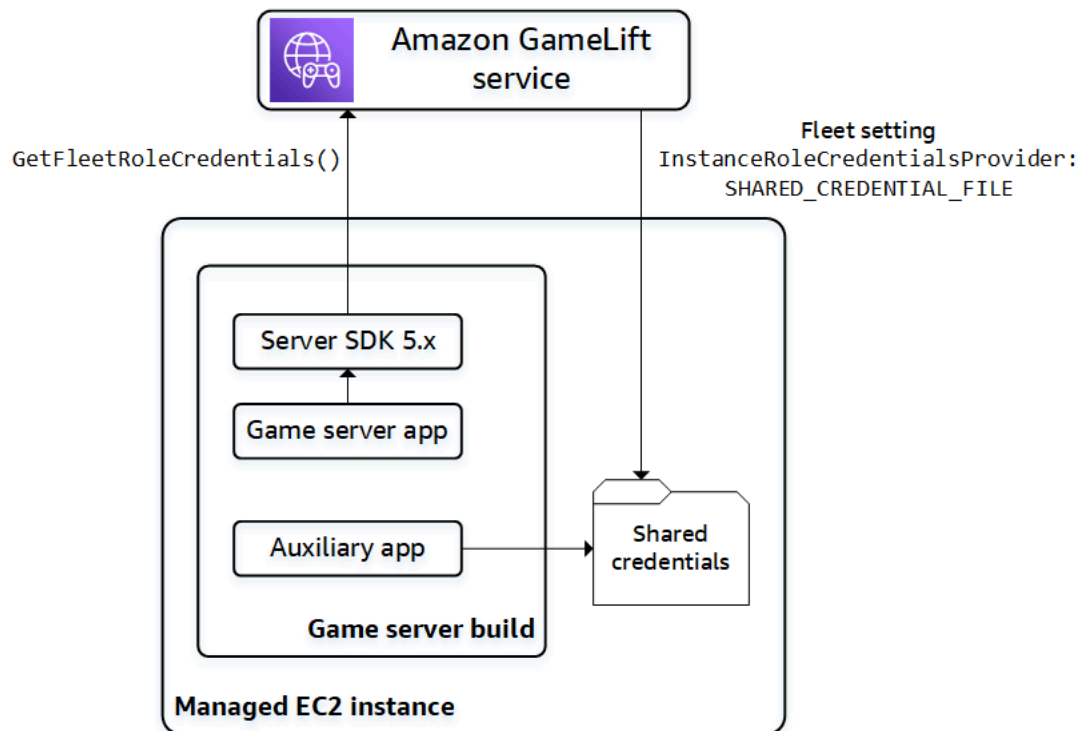
In questo passaggio, configuri le tue applicazioni per acquisire credenziali di sicurezza per il ruolo IAM e utilizzarle quando interagisci con le tue risorse. AWS Consulta la tabella seguente per determinare come modificare le applicazioni in base (1) al tipo di applicazione e (2) alla versione SDK del server con cui il gioco comunica. Amazon GameLift Servers

	Applicazioni per server di gioco	Altre applicazioni
Utilizzo della versione 5.x dell'SDK del server	Richiama il metodo SDK del server <code>GetFleetRoleCredentials()</code> dal codice del server di gioco.	Aggiungi codice all'applicazione per estrarre le credenziali da un file condiviso sull'istanza fleet.
Utilizzando la versione 4 o precedente dell'SDK del server	Chiama AWS Security Token Service (AWS STS) AssumeRole con il ruolo ARN.	Chiama AWS Security Token Service (AWS STS) AssumeRole con il ruolo ARN.

Note

Per le flotte di container, `FleetRoleArn` le credenziali vengono inserite in ogni contenitore. Le tue applicazioni possono utilizzare il provider di AWS credenziali predefinito per accedere a queste credenziali. Puoi comunque chiamare `GetFleetRoleCredentials()`, il che restituisce le stesse credenziali. Queste credenziali del ruolo della flotta sono accessibili solo all'interno dei container.

Per i giochi integrati con il server SDK 5.x, questo diagramma illustra come le applicazioni della build di gioco distribuita possono acquisire le credenziali per il ruolo IAM.



Chiamata **GetFleetRoleCredentials()** (server SDK 5.x)

Nel codice del server di gioco, che dovrebbe già essere integrato con il Amazon GameLift Servers server SDK 5.x, chiama `GetFleetRoleCredentials` ([C++](#)) ([C#](#)) ([Unreal](#)) ([Go](#)) per recuperare un set di credenziali temporanee. Quando le credenziali scadono, puoi aggiornarle con un'altra chiamata a `GetFleetRoleCredentials`.

Usa credenziali condivise (server SDK 5.x)

Per le applicazioni non server distribuite con build di server di gioco utilizzando server SDK 5.x, aggiungi codice per ottenere e utilizzare le credenziali archiviate in un file condiviso.

Amazon GameLift Servers genera un profilo di credenziali per ogni istanza del parco istanze. Le credenziali sono disponibili per l'uso da parte di tutte le applicazioni sull'istanza. Amazon GameLift Servers aggiorna continuamente le credenziali temporanee.

È necessario configurare una flotta per generare il file di credenziali condivise durante la creazione della flotta.

In ogni applicazione che deve utilizzare il file di credenziali condivise, specifica la posizione del file e il nome del profilo, come segue:

Windows:

```
[credentials]
shared_credential_profile= "FleetRoleCredentials"
shared_credential_file= "C:\\Credentials\\credentials"
```

Linux:

```
[credentials]
shared_credential_profile= "FleetRoleCredentials"
shared_credential_file= "/local/credentials/credentials"
```

Esempio: configura un CloudWatch agente per raccogliere le metriche per le istanze del parco istanze Amazon GameLift Servers

Se desideri utilizzare un CloudWatch agente Amazon per raccogliere metriche, log e tracce dalle tue Amazon GameLift Servers flotte, utilizza questo metodo per autorizzare l'agente a inviare i dati al tuo account. In questo scenario, procedi nel seguente modo:

1. Recupera o scrivi il `config.json` file CloudWatch dell'agente.
2. Aggiorna il `common-config.toml` file per consentire all'agente di identificare il nome del file delle credenziali e il nome del profilo, come descritto sopra.
3. Configura lo script di installazione della build del server di gioco per installare e avviare l'CloudWatch agente.

Usa **AssumeRole()** (server SDK 4)

Aggiungi codice alle tue applicazioni per assumere il ruolo IAM e ottenere le credenziali per interagire con le tue AWS risorse. Qualsiasi applicazione eseguita su un'istanza del Amazon GameLift Servers parco istanze con server SDK 4 o versioni precedenti può assumere il ruolo IAM.

Nel codice dell'applicazione, prima di accedere a una AWS risorsa, l'applicazione deve chiamare l'operazione [AssumeRole](#) API AWS Security Token Service (AWS STS) e specificare il ruolo ARN. Questa operazione restituisce un set di credenziali temporanee che autorizzano l'applicazione ad accedere alla risorsa. AWS Per ulteriori informazioni, consulta [Using temporary credentials with AWS resources](#) nella IAM User Guide.

Associa una flotta al ruolo IAM

Dopo aver creato il ruolo IAM e aggiornato le applicazioni nella build del server di gioco per ottenere e utilizzare le credenziali di accesso, puoi implementare una flotta. Quando configuri la nuova flotta, imposta i seguenti parametri.

Per le flotte di container:

- [FleetRoleArn](#)— Imposta questo parametro sull'ARN del ruolo IAM.

Per altri tipi di flotta:

- [InstanceRoleArn](#)— Imposta questo parametro sull'ARN del ruolo IAM.
- [InstanceRoleCredentialsProvider](#)— Amazon GameLift Servers Per richiedere la generazione di un file di credenziali condiviso per ogni istanza della flotta, imposta questo parametro su. `SHARED_CREDENTIAL_FILE`

È necessario impostare questi valori quando si crea la flotta. Non possono essere aggiornati in seguito.

Accedi alle AWS risorse con il peering VPC

Puoi utilizzare il peering di Amazon Virtual Private Cloud (Amazon VPC) per comunicare tra applicazioni in esecuzione su un'Amazon GameLift Serversistanza e un'altra risorsa. AWS Un VPC è una rete privata virtuale definita dall'utente che include un insieme di risorse gestite tramite. Account AWS Ogni parco istanze Amazon GameLift Servers ha il proprio VPC. Con il peering VPC, puoi stabilire una connessione di rete diretta tra il VPC per la tua flotta e per le altre risorse. AWS

Amazon GameLift Servers ottimizza il processo di configurazione delle connessioni VPC in peering per i server di gioco. Gestisce le richieste di peering, aggiorna le tabelle di routing e configura le connessioni in base alle esigenze. Per istruzioni su come configurare il peering VPC per i tuoi server di gioco, consulta. [Configurare il peering VPC per Amazon GameLift Servers](#)

Note

Il peering VPC non è supportato per le flotte di container.

Consenti al tuo server di gioco di accedere ai dati Amazon GameLift Servers della flotta

In alcune situazioni la build o Amazon GameLift Servers Realtime lo script del gioco personalizzato potrebbero richiedere informazioni sulla flotta. Amazon GameLift Servers Ad esempio, la build o lo script del gioco potrebbero includere codice per:

- Monitora l'attività in base ai dati della flotta.
- Aggiusta le metriche per tracciare l'attività in base ai dati della flotta. (Molti giochi utilizzano questi dati per LiveOps le attività.)
- Fornisci dati pertinenti ai servizi di gioco personalizzati, ad esempio per il matchmaking, la scalabilità aggiuntiva della capacità o i test.

Le informazioni sulla flotta sono disponibili come file JSON su ogni istanza nelle seguenti posizioni:

- Windows: C:\GameMetadata\gamelift-metadata.json
- Linux: /local/gamemetadata/gamelift-metadata.json

Il gamelift-metadata.json file include gli [attributi di una risorsa del Amazon GameLift Servers parco veicoli](#).

File JSON di esempio:

```
{
  "buildArn": "arn:aws:gamelift:us-west-2:123456789012:build/build-1111aaaa-22bb-33cc-44dd-5555eeee66ff",
  "buildId": "build-1111aaaa-22bb-33cc-44dd-5555eeee66ff",
  "fleetArn": "arn:aws:gamelift:us-west-2:123456789012:fleet/fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
  "fleetDescription": "Test fleet for Really Fun Game v0.8",
  "fleetId": "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
  "name": "ReallyFunGameTestFleet08",
  "fleetType": "ON_DEMAND",
  "instanceRoleArn": "arn:aws:iam::123456789012:role/S3AccessForGameLift",
  "instanceType": "c5.large",
  "serverLaunchPath": "/local/game/reallyfungame.exe"
}
```


Configurare il peering VPC per Amazon GameLift Servers

Questo argomento fornisce una panoramica su come impostare una connessione VPC in peering tra il server di gioco Amazon GameLift Servers in hosting e le altre risorse non Amazon GameLift Servers. Utilizza le connessioni peering Amazon Virtual Private Cloud (VPC) per consentire ai server di gioco di comunicare direttamente e privatamente con le altre AWS risorse, come un servizio Web o un repository. Puoi stabilire il peering VPC con qualsiasi risorsa in esecuzione AWS e gestita da un AWS account a cui hai accesso.

Note

Peering VPC è una caratteristica avanzata. Per maggiori informazioni sulle opzioni preferite per consentire ai server di gioco di comunicare direttamente e privatamente con le altre AWS risorse, consulta [Connect il server di gioco Amazon GameLift Servers ospitato ad altre AWS risorse](#)

Se conosci già il peering di Amazon VPCs e VPC, tieni presente che la configurazione del peering con i server di Amazon GameLift Servers gioco è leggermente diversa. Non hai accesso al VPC che contiene i tuoi server di gioco, poiché è controllato dal Amazon GameLift Servers servizio, quindi non puoi richiedere direttamente il peering VPC. Al contrario, prima preautorizzi il VPC con le risorse non Amazon GameLift Servers in modo da accettare una richiesta di peering dal servizio Amazon GameLift Servers. Successivamente attivi Amazon GameLift Servers per richiedere il peering VPC appena autorizzato. Amazon GameLift Servers gestisce le attività di creazione della connessione peering, impostazione delle tabelle di routing e configurazione della connessione.

Configurazione del peering VPC per un parco istanze esistente

1. Ottieni gli ID e le credenziali dell'account. AWS

Sono necessari un ID e le credenziali di accesso per i seguenti account. AWS Puoi trovare l' AWS account IDs accedendo [Console di gestione AWS](#) e visualizzando le impostazioni dell'account. Per ottenere le credenziali, andare alla console IAM.

- AWS account che usi per gestire i tuoi server di Amazon GameLift Servers gioco.
- AWS account che usi per gestire le tue Amazon GameLift Servers risorse diverse.

Se utilizzi lo stesso account Amazon GameLift Servers e non utilizzi Amazon GameLift Servers risorse, ti servono ID e credenziali solo per quell'account.

2. Ottenimento di identificatori per ogni VPC.

Ottieni le seguenti informazioni per VPCs consentirne il peering:

- VPC per i tuoi server di Amazon GameLift Servers gioco: questo è l'ID Amazon GameLift Servers della tua flotta. I tuoi server di gioco sono distribuiti in Amazon GameLift Servers su un parco istanze EC2. Un parco istanze viene automaticamente posizionato nel proprio VPC, gestito dal servizio Amazon GameLift Servers. Non hai accesso diretto al VPC, quindi è identificato dall'ID della flotta.
- VPC per le tue Amazon GameLift Servers AWS risorse diverse: puoi stabilire un peering VPC con tutte le risorse in esecuzione AWS e gestite da un AWS account a cui hai accesso. Se non hai ancora creato un VPC per queste risorse, consulta la [Guida introduttiva ad Amazon VPC](#). Dopo aver creato un VPC, puoi trovare l'ID VPC accedendo a [Console di gestione AWS](#) Amazon VPC e visualizzando il tuo. VPCs

Note

Quando si configura un peering, entrambi VPCs devono esistere nella stessa regione. Il VPC per i server di gioco del parco istanze Amazon GameLift Servers si trova nella stessa regione del parco istanze.

3. Autorizza un peering VPC.

In questa fase pre-autorizzi una richiesta futura di Amazon GameLift Servers per eseguire il peering del VPC con i server di gioco con il VPC per risorse non Amazon GameLift Servers. Questa operazione aggiorna il gruppo di sicurezza per il VPC.

Per autorizzare il peering VPC, chiama il service [CreateVpcPeeringAuthorizationAPI \(\)](#) o usa il comando CLI AWS `create-vpc-peering-authorization` Effettua questa chiamata utilizzando l'account che gestisce le risorse non Amazon GameLift Servers. Identificare le informazioni che seguono:

- ID VPC peer: è per il VPC che non dispone di risorse. Amazon GameLift Servers

- Amazon GameLift Servers AWS ID account: questo è l'account che usi per gestire la tua flotta. Amazon GameLift Servers

Una volta autorizzato un peering VPC, l'autorizzazione rimane valida per 24 ore a meno che non venga revocata. È possibile gestire le autorizzazioni peer VPC utilizzando le operazioni descritte di seguito:

- [DescribeVpcPeeringAuthorizations\(\)](#) (AWS CLI `describe-vpc-peering-authorizations`).
- [DeleteVpcPeeringAuthorization\(\)](#) (AWS CLI `delete-vpc-peering-authorization`).

4. Richiedi una connessione peering.

Con un'autorizzazione valida, è possibile richiedere che Amazon GameLift Servers stabilisca una connessione peering.

Per richiedere un peering VPC, chiama il service API [CreateVpcPeeringConnection\(\)](#) o usa il comando AWS CLI `create-vpc-peering-connection`. Effettua questa chiamata utilizzando l'account che gestisce i server di gioco Amazon GameLift Servers. Utilizza le seguenti informazioni per identificare i due VPCs che desideri eseguire il peering:

- ID VPC peer AWS e ID account: questo è il VPC per le tue non Amazon GameLift Servers risorse e l'account che usi per gestirle. L'ID VPC deve corrispondere all'ID di un'autorizzazione peering valida.
- Fleet ID: identifica il VPC per i Amazon GameLift Servers tuoi server di gioco.

5. Monitora lo stato della connessione peering.

Richiedere una connessione peer VPC è un'operazione asincrona. Per monitorare lo stato di una richiesta di peer e gestire i casi di successo o di errore, utilizzare una delle seguenti opzioni:

- Effettuare continuamente il polling con `DescribeVpcPeeringConnections()`. Questa operazione recupera il record di connessione peer VPC, tra cui lo stato della richiesta. Se una connessione peer viene creata correttamente, il record di connessione contiene anche un blocco CIDR di indirizzi IP privati assegnato al VPC.
- Gestisci gli eventi della flotta associati alle connessioni di peering VPC con [DescribeFleetEvents\(\)](#), inclusi gli eventi di successo e di fallimento.

Una volta stabilita la connessione peering, puoi gestirla utilizzando le seguenti operazioni:

- [DescribeVpcPeeringConnections\(\)](#) (AWS CLI `describe-vpc-peering-connections`).
- [DeleteVpcPeeringConnection\(\)](#) (AWS CLI `delete-vpc-peering-connection`).

Per configurare il peering VPC con un nuovo parco istanze

Puoi creare un nuovo parco istanze Amazon GameLift Servers e richiedere contemporaneamente una connessione peering VPC.

1. Ottieni gli ID e le credenziali dell' AWS account.

Sono necessari un ID e le credenziali di accesso per i due account seguenti. AWS Puoi trovare l' AWS account IDs accedendo [Console di gestione AWS](#) e visualizzando le impostazioni dell'account. Per ottenere le credenziali, andare alla console IAM.

- AWS account che usi per gestire i tuoi server di Amazon GameLift Servers gioco.
- AWS account che usi per gestire le tue Amazon GameLift Servers risorse diverse.

Se utilizzi lo stesso account Amazon GameLift Servers e non utilizzi Amazon GameLift Servers risorse, ti servono ID e credenziali solo per quell'account.

2. Ottieni l'ID VPC per le tue risorse diverse. Amazon GameLift Servers AWS

Se non hai ancora creato un VPC per queste risorse, fallo ora (vedi [Guida introduttiva ad Amazon VPC](#)). Assicurati di creare il nuovo VPC nella stessa regione in cui prevedi di creare il nuovo parco istanze. Se le tue Amazon GameLift Servers risorse non sono gestite con un AWS account o un user/user gruppo diverso da quello con cui utilizzi Amazon GameLift Servers, dovrai utilizzare queste credenziali dell'account quando richiedi l'autorizzazione nel passaggio successivo.

Dopo aver creato un VPC, puoi individuare l'ID VPC nella console Amazon VPC visualizzando il tuo. VPCs

3. Autorizzazione di un VPC in peering con risorse non Amazon GameLift Servers.

Quando Amazon GameLift Servers crea il nuovo parco istanze e il VPC corrispondente, invia anche una richiesta di peering con il VPC per le risorse non Amazon GameLift Servers. Devi preautorizzare la richiesta. Questa fase aggiorna il gruppo di sicurezza per il VPC.

Utilizzando le credenziali dell'account che gestiscono le tue Amazon GameLift Servers risorse diverse, chiama l'API del servizio [CreateVpcPeeringAuthorization\(\)](#) o usa il comando AWS `create-vpc-peering-authorization` CLI. Identificare le informazioni che seguono:

- Peer VPC ID: ID del VPC con le tue risorse non disponibili. Amazon GameLift Servers
- Amazon GameLift Servers AWS ID account: ID dell'account che utilizzi per gestire la tua flotta. Amazon GameLift Servers

Una volta autorizzato un peering VPC, l'autorizzazione rimane valida per 24 ore a meno che non venga revocata. È possibile gestire le autorizzazioni peer VPC utilizzando le operazioni descritte di seguito:

- [DescribeVpcPeeringAuthorizations\(\)](#) (AWS CLI `describe-vpc-peering-authorizations`).
 - [DeleteVpcPeeringAuthorization\(\)](#) (AWS CLI `delete-vpc-peering-authorization`).
4. Segui le istruzioni per [creare una nuova flotta utilizzando la AWS CLI](#). Includere i seguenti parametri aggiuntivi:
- `peer-vpc-aws-account-id` — ID dell'account che usi per gestire il VPC con le tue non Amazon GameLift Servers risorse.
 - `peer-vpc-id` — ID del VPC con il tuo account non registrato. Amazon GameLift Servers

Una chiamata effettuata correttamente a [create-fleet](#) con i parametri di peering VPC genera un nuovo parco istanze e una nuova richiesta di peering VPC. Lo stato del parco istanze è impostato su New (Nuovo) e il processo di attivazione del parco istanze viene avviato. Lo stato della richiesta di connessione peering è impostato su `initiating-request`. È possibile monitorare l'esito positivo o negativo della richiesta di peering chiamando. [describe-vpc-peering-connections](#)

Quando richiedi un nuovo parco istanze e una connessione peering VPC, entrambe le operazioni possono riuscire correttamente o generare un errore. Se un parco istanze genera un errore durante il processo di creazione, la connessione VPC in peering non verrà stabilita. Analogamente, se una connessione peering VPC non riesce per qualsiasi motivo, il nuovo parco istanze non potrà passare dallo stato `Activating` (In attivazione) allo stato `Active` (Attivo).

Note

La nuova connessione peering VPC non viene completata finché il parco istanze non è pronto per l'attivazione. Ciò significa che la connessione non è disponibile e non può essere utilizzata durante il processo di installazione della build del server di gioco.

Nell'esempio seguente viene creato un nuovo parco istanze e una connessione peering tra un VPC prestabilito e il VPC per il nuovo parco istanze. Il VPC prestabilito viene identificato in modo univoco dalla combinazione del tuo ID non Amazon GameLift Servers AWS account e dell'ID VPC.

```
$ AWS gamelift create-fleet
  --name "My_Fleet_1"
  --description "The sample test fleet"
  --ec2-instance-type "c5.large"
  --fleet-type "ON_DEMAND"
  --build-id "build-1111aaaa-22bb-33cc-44dd-5555eeee66ff"
  --runtime-configuration "GameSessionActivationTimeoutSeconds=300,
                           MaxConcurrentGameSessionActivations=2,
                           ServerProcesses=[{LaunchPath=C:\game\Bin64.dedicated
\MultiplayerSampleProjectLauncher_Server.exe,
                                           Parameters+=sv_port 33435 +start_lobby,
                                           ConcurrentExecutions=10}]]"
  --new-game-session-protection-policy "FullProtection"
  --resource-creation-limit-policy "NewGameSessionsPerCreator=3,
                                   PolicyPeriodInMinutes=15"
  --ec2-inbound-permissions
  "FromPort=33435,ToPort=33435,IpRange=0.0.0.0/0,Protocol=UDP"
  "FromPort=33235,ToPort=33235,IpRange=0.0.0.0/0,Protocol=UDP"
  --metric-groups "EMEAfleets"
  --peer-vpc-aws-account-id "111122223333"
  --peer-vpc-id "vpc-a11a11a"
```

Versione copiabile:

```
AWS gamelift create-fleet --name "My_Fleet_1" --description "The
sample test fleet" --fleet-type "ON_DEMAND" --metric-groups
"EMEAfleets" --build-id "build-1111aaaa-22bb-33cc-44dd-5555eeee66ff"
--ec2-instance-type "c5.large" --runtime-configuration
"GameSessionActivationTimeoutSeconds=300,MaxConcurrentGameSessionActivations=2,ServerProcesses
```

```
\game\Bin64.dedicated\MultiplayerSampleProjectLauncher_Server.exe,Parameters=
+sv_port 33435 +start_lobby,ConcurrentExecutions=10}]" --new-game-session-
protection-policy "FullProtection" --resource-creation-limit-policy
"NewGameSessionsPerCreator=3,PolicyPeriodInMinutes=15" --ec2-inbound-
permissions "FromPort=33435,ToPort=33435,IpRange=0.0.0.0/0,Protocol=UDP"
"FromPort=33235,ToPort=33235,IpRange=0.0.0.0/0,Protocol=UDP" --peer-vpc-aws-account-id
"111122223333" --peer-vpc-id "vpc-a11a11a"
```

Risoluzione dei problemi di peering VPC

Se hai problemi a stabilire una connessione peering VPC per i tuoi server di gioco Amazon GameLift Servers, considera le seguenti cause principali comuni:

- Non è stata trovata alcuna autorizzazione per la connessione richiesta:
 - Controlla lo stato di un'autorizzazione VPC per il VPC non Amazon GameLift Servers. Potrebbe non esistere o essere scaduto.
 - Controlla le regioni delle due regioni che VPCs stai cercando di peerizzare. Se non si trovano nella stessa regione, non possono essere sottoposti al peering.
- I blocchi CIDR (vedi [Configurazioni di connessione peering VPC non valide](#)) dei due blocchi si sovrappongono. VPCs I blocchi IPv4 CIDR assegnati al peered non possono sovrapporsi. VPCs Il blocco CIDR del VPC per la Amazon GameLift Servers tua flotta viene assegnato automaticamente e non può essere modificato, quindi dovrai modificare il blocco CIDR del VPC per le tue risorse diverse. Amazon GameLift Servers Per risolvere il problema:
 - Cerca questo blocco CIDR per il parco istanze Amazon GameLift Servers chiamando `DescribeVpcPeeringConnections()`.
 - Vai alla console Amazon VPC, trova il VPC per le risorse non Amazon GameLift Servers e modifica il blocco CIDR in modo che non si sovrapponga.
- Il nuovo parco istanze non è attivato (quando si richiede il peering VPC con un nuovo parco istanze). Se il nuovo parco istanze non è riuscito a passare allo stato Attivo non vi è alcun VPC da sottoporre al peering, pertanto la connessione peering non può avere esito positivo.

Sessioni dei giocatori e personalizzazioni del matchmaking

Le personalizzazioni delle sessioni dei giocatori e del matchmaking ti danno l'opportunità di sviluppare sofisticati flussi di lavoro di gestione dei giocatori, incluso un sistema di matchmaking ricco di sfumature che ti aiuta a offrire esperienze multigiocatore equilibrate e coinvolgenti.

Genera giocatore IDs

Amazon GameLift Servers utilizza una sessione giocatore per rappresentare un giocatore connesso a una sessione di gioco. Amazon GameLift Servers crea una sessione di gioco ogni volta che un giocatore si connette a una sessione di gioco utilizzando un client di gioco integrato con Amazon GameLift Servers. Quando un giocatore lascia una partita, la sessione del giocatore termina. Amazon GameLift Servers non riutilizza le sessioni dei giocatori.

Important

Quando usi il FlexMatch matchmaking, se crei una nuova richiesta di matchmaking contenente un ID giocatore già incluso in una richiesta di matchmaking attiva esistente, la richiesta esistente viene automaticamente annullata. Tuttavia, non viene inviato un MatchmakingCancelled evento per la richiesta annullata. Per monitorare lo stato delle richieste di matchmaking esistenti, usa [DescribeMatchmaking](#) per controllare lo stato della richiesta a intervalli non frequenti (30-60 secondi). La richiesta annullata mostrerà lo stato di con il motivo. CANCELLED Cancelled due to duplicate player

Il seguente esempio di codice genera casualmente un giocatore unico: IDs

```
bool includeBrackets = false;
bool includeDashes = true;
string playerId = AZ::Uuid::CreateRandom().ToString<string>(includeBrackets,
    includeDashes);
```

Per ulteriori informazioni sulle sessioni con i giocatori, consulta. [Sessioni di gioco e di gioco nella Amazon GameLift Servers console](#)

Aggiungi FlexMatch matchmaking a Amazon GameLift Servers

Utilizzalo Amazon GameLift Servers FlexMatch per aggiungere funzionalità di matchmaking tra giocatori ai giochi Amazon GameLift Servers ospitati. Puoi utilizzarlo FlexMatch con server di gioco personalizzati o Amazon GameLift ServersRealtime.

FlexMatch associa il servizio di abbinamento a un motore di regole personalizzabili. Progetti come abbinare i giocatori in base agli attributi dei giocatori e alle modalità di gioco più adatte al tuo gioco. FlexMatch gestisce le fasi di valutazione dei giocatori che stanno cercando una partita, organizza le partite con una o più squadre e avvia le sessioni di gioco per ospitare le partite.

Per utilizzare il FlexMatch servizio completo, è necessario che le risorse di hosting siano configurate con code. Amazon GameLift Servers utilizza le code per individuare le migliori sedi di hosting possibili per i giochi in diverse regioni e tipi di computer. In particolare, le Amazon GameLift Servers code possono utilizzare i dati di latenza, se forniti dai client di gioco, per organizzare le sessioni di gioco in modo che i giocatori abbiano la latenza più bassa possibile durante il gioco.

[Per ulteriori informazioni su FlexMatch come includere assistenza dettagliata sull'integrazione del matchmaking nei tuoi giochi, consulta questi argomenti della Guida per gli sviluppatori: Amazon GameLift Servers FlexMatch](#)

- [Come funziona Amazon GameLift Servers FlexMatch](#)
- [FlexMatch fasi di integrazione](#)

Personalizzazioni del posizionamento delle sessioni di gioco

Le personalizzazioni del posizionamento delle sessioni di gioco ti danno l'opportunità di perfezionare il tuo sistema di posizionamento per offrire alla tua base di giocatori la migliore esperienza di gioco possibile. Puoi ottimizzare il posizionamento in base alla compatibilità e alle preferenze dei giocatori, nonché per considerazioni operative come l'efficienza dei costi, la distribuzione geografica e la resilienza contro le interruzioni del servizio.

Personalizza una coda di sessioni di gioco

Questo argomento descrive come personalizzare le code delle sessioni di gioco per prendere le migliori decisioni possibili sul posizionamento delle sessioni di gioco. Per ulteriori informazioni sulle code delle sessioni di gioco e sul loro funzionamento, consulta. [Configura il posizionamento delle sessioni di gioco](#)

Queste Amazon GameLift Servers funzionalità richiedono code:

- [Abbinamento con FlexMatch](#)
- [Crea una coda per le istanze Spot](#)

Argomenti

- [Definire l'ambito di una coda](#)
- [Crea una coda con più sedi](#)
- [Valuta le metriche della coda](#)

Definire l'ambito di una coda

La popolazione di giocatori del tuo gioco potrebbe avere gruppi di giocatori che non dovrebbero giocare insieme. Ad esempio, se pubblichi il gioco in due lingue, ogni lingua dovrebbe avere i propri server di gioco.

Per impostare il posizionamento delle sessioni di gioco in base al numero di giocatori, crea una coda separata per ogni segmento di giocatori. Analizza ogni coda per posizionare i giocatori nei server di gioco corretti. Alcuni metodi comuni per definire l'ambito delle code includono:

- Per località geografiche. Quando distribuisce i tuoi server di gioco in più aree geografiche, potresti creare code per i giocatori in ogni località per ridurre la latenza dei giocatori.
- Tramite build o varianti dello script. Se hai più di una variante del tuo server di gioco, potresti supportare gruppi di giocatori che non possono giocare nelle stesse sessioni di gioco. Ad esempio, le build o gli script dei server di gioco potrebbero supportare lingue o tipi di dispositivi diversi.
- Per tipi di eventi. Puoi creare una coda speciale per gestire le partite per i partecipanti a tornei o altri eventi speciali.

Progetta code multiple

A seconda del gioco e dei giocatori, potresti voler creare più di una coda per le sessioni di gioco. Quando il servizio client di gioco richiede una nuova sessione di gioco, specifica quale coda di sessione di gioco utilizzare. Per aiutarti a determinare se utilizzare più code, considera:

- Varianti del tuo server di gioco. Puoi creare una coda separata per ogni variante del tuo server di gioco. Tutte le flotte in coda devono utilizzare server di gioco compatibili. Questo perché i giocatori che utilizzano la coda per partecipare alle partite devono poter giocare su qualsiasi server di gioco della coda.
- Gruppi di giocatori diversi. Puoi personalizzare Amazon GameLift Servers il posizionamento delle sessioni di gioco in base al gruppo di giocatori. Ad esempio, potresti aver bisogno di code personalizzate per determinate modalità di gioco che richiedono un tipo di istanza o una configurazione di runtime speciali. Oppure, potresti aver bisogno di una coda speciale per gestire i piazzamenti per un torneo o un altro evento.
- Metriche della coda delle sessioni di gioco. Puoi impostare le code in base a come desideri raccogliere le metriche di posizionamento delle sessioni di gioco. Per ulteriori informazioni, consulta [Parametri Amazon GameLift Servers per code](#).

Crea una coda con più sedi

Consigliamo una progettazione con più posizioni per tutte le code. Questo design può migliorare la velocità di posizionamento e la resilienza dell'hosting. È necessario un design multi-location per utilizzare i dati di latenza dei giocatori per coinvolgere i giocatori in sessioni di gioco con una latenza minima. Se stai creando code con più sedi che utilizzano flotte di istanze Spot, segui le istruzioni riportate in [Riduci i costi di hosting dei giochi con le flotte Spot](#)

Un modo per creare una coda con più sedi consiste nell'aggiungere un parco veicoli con [più](#) sedi a una coda. In questo modo, la coda può collocare sessioni di gioco in qualsiasi postazione della flotta. Puoi anche aggiungere altre flotte con configurazioni diverse o ubicazioni principali per ridondarle. Se utilizzi una flotta di istanze Spot con più sedi, segui le best practice e includi una flotta di istanze On-Demand con le stesse sedi.

L'esempio seguente illustra il processo di progettazione di una coda di base con più sedi. In questo esempio, utilizziamo due flotte: una flotta di istanze Spot e una flotta di istanze On-Demand. Ogni flotta ha le seguenti località Regioni AWS di collocamento: `us-east-1`, `us-east-2`, `ca-central-1`, e `us-west-2`

Per creare una coda di base con più sedi con flotte con più sedi

1. Scegli una posizione in cui creare la coda. È possibile ridurre al minimo la latenza delle richieste posizionando la coda in una posizione vicina a dove è stato distribuito il servizio client. In questo esempio, creiamo la coda in `us-east-1`
2. Crea una nuova coda e aggiungi le tue flotte con più sedi come destinazioni di coda. L'ordine di destinazione determina come Amazon GameLift Servers posiziona le sessioni di gioco. In questo esempio, elenchiamo prima il parco istanze Spot e poi il parco istanze On-Demand.
3. Definisci l'ordine di priorità di posizionamento delle sessioni di gioco della coda. Questo ordine determina dove la coda cerca per primo un server di gioco disponibile. In questo esempio, utilizziamo l'ordine di priorità predefinito.
4. Definire l'ordine delle ubicazioni. Se non definisci l'ordine delle località, Amazon GameLift Servers utilizza le posizioni in ordine alfabetico.

Game session placement locations

Locations where the queue can place new game sessions.

Locations

Choose locations ▼

ca-central-1
Canada (Central) ✕

us-west-2
US West (Oregon) ✕

us-east-2
US East (Ohio) ✕

us-east-1
US East (N. Virginia) ✕

Destination order

An ordered list of fleets and aliases that the queue can use for game session placement.

	Region	Type	Name	
⋮	us-east-1 ▼	Fleet ▼	TestFleet-SPOT ▼	Remove
⋮	us-east-1 ▼	Fleet ▼	TestFleet-ONDEMAND ▼	Remove
Add Destination				

Game session placement priority

The values that GameLift uses to prioritize game session placement. The default order is latency, cost, destination, and location.

⋮

Latency

Prioritize locations with the lowest average player latency.

⋮

Cost

Prioritize destinations with the lowest current hosting cost.

⋮

Destination

Prioritize based on the defined destination order.

⋮

Location

Prioritize based on the defined location order.

▼

Location order

An ordered list of locations that the queue can use for game session placement.

Location

ca-central-1

▼

Remove

us-east-1

▼

Remove

us-east-2

▼

Remove

us-west-2

▼

Remove

Add locations

Valuta le metriche della coda

I parametri consentono di valutare le prestazioni delle code. È possibile visualizzare le metriche relative alle code in [Amazon GameLift Servers console](#) o in Amazon CloudWatch. Per un elenco e le descrizioni delle metriche delle code, consulta. [Parametri Amazon GameLift Servers per code](#)

Le metriche della coda possono fornire informazioni su quanto segue:

- **Prestazioni complessive della coda:** le metriche della coda indicano il successo con cui una coda risponde alle richieste di posizionamento. Queste metriche possono anche aiutarti a identificare quando e perché i posizionamenti falliscono. Per le code con flotte ridimensionate manualmente, le QueueDepth metriche AverageWaitTime e possono indicare quando è necessario regolare la capacità di una coda.
- **FleetIQ prestazioni dell'algoritmo:** per le richieste di posizionamento utilizzando il FleetIQ algoritmo, le metriche mostrano la frequenza con cui l'algoritmo trova il posizionamento ideale nella sessione di gioco. Il posizionamento può dare la priorità all'utilizzo di risorse con la latenza di giocatore più bassa o di risorse con il costo più basso. Esistono anche metriche di errore che identificano i motivi più comuni Amazon GameLift Servers non riesco a trovare un posizionamento ideale. Per ulteriori informazioni sui parametri, consulta [Monitora Amazon GameLift Servers con Amazon CloudWatch](#).
- **Posizionamenti specifici per località:** per le code con più sedi, le metriche mostrano i posizionamenti riusciti per località. Per le code che utilizzano il FleetIQ algoritmo, questi dati forniscono informazioni utili su dove avviene l'attività dei giocatori.

Quando si valutano le metriche per FleetIQ prestazioni dell'algoritmo, considera i seguenti suggerimenti:

- Per tenere traccia del tasso di ricerca del posizionamento ideale da parte della coda, utilizza la PlacementsSucceeded metrica in combinazione con FleetIQ metriche per la latenza più bassa e il prezzo più basso.
- Per aumentare il tasso di ricerca del posizionamento ideale da parte di una coda, esamina le seguenti metriche di errore:
 - Se il valore FirstChoiceOutOfCapacity è elevato, modifica la scalabilità della capacità per le flotte della coda.
 - Se la metrica FirstChoiceNotViable di errore è alta, dai un'occhiata alle flotte delle tue istanze Spot. Le flotte di istanze Spot sono considerate inutilizzabili quando il tasso di interruzione per un particolare tipo di istanza è troppo elevato. Per risolvere questo problema, modifica la coda per utilizzare flotte di istanze Spot con diversi tipi di istanze. Ti consigliamo di includere flotte di istanze Spot con diversi tipi di istanze in ogni sede.

Dare la priorità al posizionamento delle sessioni di gioco

Amazon GameLift Servers utilizza un algoritmo per determinare come dare priorità alle destinazioni di una coda e determinare dove collocare una nuova sessione di gioco. L'algoritmo si basa su un

insieme ordinato di criteri. Puoi utilizzare l'ordine di priorità predefinito oppure puoi personalizzare l'ordine. È possibile modificare l'ordine di priorità di una coda in qualsiasi momento.

Ordine di priorità predefinito

1. **Latenza:** se la richiesta di posizionamento della sessione di gioco include dati di latenza specifici per località per i giocatori, Amazon GameLift Servers calcola la latenza media dei giocatori in ogni località e tenta di piazzare una sessione di gioco in una località della flotta con la media più bassa.
2. **Costo:** se una richiesta non include dati sulla latenza o se più flotte hanno la stessa latenza, valuta il costo di hosting di ciascuna flotta. Amazon GameLift Servers Il costo di hosting di una flotta varia in base al tipo di flotta (Spot o On-Demand), al tipo di istanza e alla posizione.
3. **Destinazione:** se più flotte hanno latenza e costi uguali, assegna la Amazon GameLift Servers priorità alle flotte in base all'ordine di destinazione indicato nella configurazione della coda.
4. **Ubicazione:** per le code con flotte con più sedi, se tutti gli altri criteri sono uguali, assegna la Amazon GameLift Servers priorità alle posizioni della flotta in base all'ordine alfabetico.

Personalizza il modo in cui una coda assegna la priorità ai posizionamenti delle sessioni di gioco

Puoi scegliere di personalizzare il modo in cui una coda assegna la priorità ai criteri di posizionamento. La coda applica la priorità personalizzata a tutte le richieste di posizionamento delle sessioni di gioco che riceve.

Note

Se crei una configurazione di priorità personalizzata e non includi tutti e quattro i criteri, aggiunge Amazon GameLift Servers automaticamente i criteri mancanti nell'ordine predefinito.

Per personalizzare la configurazione della priorità di una coda

Usa la [Amazon GameLift Serversconsole](#) o AWS Command Line Interface (AWS CLI) per creare una configurazione di priorità personalizzata.

Console

Nella [Amazon GameLift Serversconsole](#), è possibile personalizzare le priorità di una coda quando si crea una nuova coda o si aggiorna una coda esistente. Seleziona una AWS regione in cui lavorare.

Apri la barra di navigazione a sinistra della console e scegli Code. Nella pagina Code, seleziona una coda esistente e scegli Modifica.

1. Vai alla sezione Priorità del posizionamento della sessione di gioco. Trascina e rilascia ogni criterio di priorità per creare l'ordine desiderato.
2. Vai alla sezione Ordine di ubicazione. Aggiungi tutte le sedi a cui desideri dare la priorità. Questo elenco è utile quando la coda ha flotte con più sedi. È necessario specificare almeno una posizione. Alle posizioni specificate qui viene data priorità per prime, seguite da tutte le altre posizioni nelle destinazioni della coda.
3. Scegli Save changes (Salva modifiche).

AWS CLI

Utilizzate il [update-game-session-queue](#) comando con l'`--priority-configuration` opzione per personalizzare l'ordine di priorità di una coda. Amazon GameLift Servers aggiorna una coda nella AWS regione predefinita corrente oppure puoi aggiungere un `--region` tag per specificare una regione diversa AWS .

La seguente richiesta di esempio aggiunge o aggiorna la configurazione di priorità per una coda specificata

```
aws gamelift update-game-session-queue \  
  --name "example-queue-with-priority" \  
  --priority-configuration \  
    PriorityOrder="COST", "LOCATION", "DESTINATION", LocationOrder="us-east-1", "us- \  
east-2", "ca-central-1", "us-west-2" \  

```

Dai priorità ai posizionamenti in base alla latenza del giocatore

Se vuoi offrire ai tuoi giocatori la migliore esperienza di gioco possibile e garantire una latenza minima, procedi nel seguente modo durante la configurazione del sistema di posizionamento delle sessioni di gioco:

- Imposta la coda per dare priorità alla latenza quando scegli dove collocare le sessioni di gioco. Per impostazione predefinita, la latenza è in cima all'elenco delle priorità. Puoi anche personalizzare la configurazione delle priorità della coda e scegliere dove mettere la latenza in ordine di priorità.
- Imposta le politiche di latenza dei giocatori per la tua coda. Le politiche di latenza ti consentono di impostare limiti rigidi sulla quantità di latenza da consentire nel posizionamento di una sessione di

gioco. Se non Amazon GameLift Servers riesci a completare una sessione di gioco senza superare i limiti, la richiesta di posizionamento scadrà e avrà esito negativo. Puoi impostare un'unica politica di latenza oppure puoi creare una serie di politiche che riducono gradualmente il limite di latenza nel tempo. Con una serie di policy, puoi specificare limiti di latenza iniziale molto bassi e comunque soddisfare i giocatori con latenze più elevate dopo un breve periodo di tempo. Per i dettagli sulla creazione di politiche di latenza, consulta. [Crea una politica di latenza dei giocatori](#)

- Quando effettui richieste di posizionamento delle sessioni di gioco (vedi [StartGameSessionPlacement](#)), includi i dati sulla latenza per ogni giocatore. I dati sulla latenza dei giocatori includono un valore per ogni possibile posizione in cui potrebbe essere tenuta una sessione di gioco. Ad esempio, per una coda che colloca le sessioni di gioco in Regioni AWS us-east-2 e ca-central-1, i dati di latenza potrebbero essere simili ai seguenti:

```
"PlayerLatencies": [  
  { "LatencyInMilliseconds": 100, "PlayerId": "player1", "RegionIdentifier": "us-east-2" },  
  { "LatencyInMilliseconds": 100, "PlayerId": "player1", "RegionIdentifier": "ca-central-1" },  
  { "LatencyInMilliseconds": 150, "PlayerId": "player2", "RegionIdentifier": "us-east-2" },  
  { "LatencyInMilliseconds": 150, "PlayerId": "player2", "RegionIdentifier": "ca-central-1" }  
]
```

Per ottenere misurazioni accurate della latenza, utilizzate i beacon ping UDP. Amazon GameLift Servers Questi endpoint consentono di misurare l'effettiva latenza della rete UDP tra i dispositivi dei giocatori e ciascuna delle potenziali sedi di hosting, con conseguenti decisioni di posizionamento più accurate rispetto all'utilizzo dei ping ICMP. Per ulteriori informazioni sull'utilizzo dei beacon ping UDP per misurare la latenza, fare riferimento a. [UDP ping beacon](#)

Assegna priorità ai posizionamenti in base alla posizione

Puoi configurare una coda per posizionare le sessioni di gioco in base a un elenco di posizioni geografiche con priorità. La posizione è uno dei criteri che determinano il modo in cui una coda sceglie dove collocare una nuova sessione di gioco. Per impostazione predefinita, la posizione ha la priorità al quarto posto, dopo latenza, costo e destinazione.

Per quanto riguarda il posizionamento della sessione di gioco, destinazione e luogo hanno significati leggermente diversi:

- La destinazione si riferisce a una flotta specifica e include tutte le risorse di hosting della flotta, ovunque siano dispiegate. Quando stabilisci le priorità in base alla destinazione, Amazon GameLift Servers potresti effettuare un posizionamento in qualsiasi località della flotta. Le flotte gestite da più sedi e le flotte Anywhere possono disporre di risorse di hosting distribuite in una o più sedi.
- La posizione si riferisce a una posizione geografica specifica in cui vengono distribuite le risorse di hosting di una flotta. Una flotta può avere più sedi Regioni AWS, tra cui Local Zones o sedi personalizzate (per una flotta Anywhere). Una flotta gestita da un'unica sede ha un'unica sede ed è sempre una Regione AWS. Una flotta gestita da più sedi ha una regione di origine e può avere sedi remote. Una flotta Anywhere ha una o più sedi personalizzate.

Quando assegna la priorità ai posizionamenti in base alla località, Amazon GameLift Servers cerca tutte le destinazioni in coda che includono la posizione prioritaria e cerca in esse una risorsa di hosting disponibile. Se ci sono più destinazioni con la posizione prioritaria, Amazon GameLift Servers passa ai criteri di priorità successivi (costo, latenza, destinazione).

Esistono diversi modi per influenzare il modo in cui viene assegnata la priorità alle posizioni di una coda

- Configura il modo in cui la coda gestisce tutte le richieste di posizionamento delle sessioni di gioco:
 - Aggiungi una configurazione prioritaria alla coda. La configurazione prioritaria di una coda include un elenco ordinato di posizioni. È possibile specificare una o più posizioni a cui dare la priorità. Questo elenco non esclude alcuna località, indica semplicemente Amazon GameLift Servers dove cercare prima una risorsa di hosting disponibile. Un uso comune di un elenco di sedi ordinato è quando si desidera incanalare la maggior parte del traffico verso una o più località geografiche specifiche e utilizzare posizioni aggiuntive come capacità di backup. Aggiungi una configurazione prioritaria [UpdateGameSessionQueue](#) chiamando.
 - Aggiungi una configurazione di filtro alla coda. Una configurazione di filtro è un elenco di elementi consentiti per la coda. Indica Amazon GameLift Servers di ignorare tutte le posizioni che non sono nell'elenco quando si cerca una risorsa di hosting disponibile. Esistono due usi comuni per la configurazione di un filtro. Innanzitutto, per le flotte con più sedi, potresti utilizzare un filtro per escludere alcune sedi della flotta. In secondo luogo, potresti voler impedire temporaneamente i posizionamenti in una determinata località; ad esempio, una località potrebbe presentare problemi transitori. Poiché è possibile aggiornare la configurazione dei filtri di una coda in qualsiasi momento, è possibile aggiungere e rimuovere facilmente le posizioni in base alle esigenze. Aggiungi una configurazione di filtro [UpdateGameSessionQueue](#) chiamando.

- Utilizza istruzioni speciali per le richieste di collocamento individuali:
 - Includi un elenco di priorità prioritarie in una richiesta di posizionamento per una sessione di gioco. Puoi fornire un elenco alternativo di sedi con priorità per qualsiasi [StartGameSessionPlacement](#) richiesta. Questo elenco sostituisce efficacemente la prioritizzazione configurata della coda per le posizioni solo per quella richiesta. Non ha alcun impatto su altre richieste. Questa funzionalità di override ha alcuni requisiti:
 - Usa un elenco di override solo con una coda che ha una configurazione di priorità impostata LOCATION come prima priorità.
 - Non includere i dati sulla latenza dei giocatori nella stessa richiesta di posizionamento. L'inclusione dei dati sulla latenza crea conflitti quando si assegnano priorità a posizioni che Amazon GameLift Servers non possono essere risolte.
 - Decidi come procedere se non riesce Amazon GameLift Servers a trovare una risorsa disponibile nell'elenco delle eccezioni di priorità. Scegli se tornare alle altre posizioni della coda o limitare i posizionamenti all'elenco delle eccezioni. Per impostazione predefinita, Amazon GameLift Servers torna indietro per tentare di posizionarlo nelle altre posizioni della coda.
 - Aggiorna la configurazione dei filtri della coda in base alle esigenze, ad esempio aggiungendo posizioni nell'elenco delle sostituzioni. L'elenco delle eccezioni non invalida l'elenco dei filtri.

Crea una politica di latenza dei giocatori

Se le tue richieste di posizionamento includono dati sulla latenza dei giocatori, Amazon GameLift Servers trova le sessioni di gioco nelle località con la latenza media più bassa per tutti i giocatori. Il posizionamento delle sessioni di gioco in base alla latenza media dei giocatori Amazon GameLift Servers impedisce di collocare la maggior parte dei giocatori in partite con latenza elevata. Tuttavia, posiziona Amazon GameLift Servers ancora i giocatori con una latenza estrema. Per soddisfare questi giocatori, crea delle politiche sulla latenza dei giocatori.

Una politica di latenza dei giocatori Amazon GameLift Servers impedisce di collocare una sessione di gioco richiesta in qualsiasi punto in cui i giocatori inclusi nella richiesta potrebbero riscontrare una latenza superiore al valore massimo. Le politiche di latenza dei giocatori possono anche Amazon GameLift Servers impedire di abbinare le richieste di sessione di gioco ai giocatori con latenza più elevata.

 Tip

Per gestire regole specifiche sulla latenza, ad esempio richiedere una latenza simile per tutti i giocatori di un gruppo, puoi utilizzare la funzione [Amazon GameLift ServersFlexMatch](#) per creare regole di matchmaking basate sulla latenza.

Ad esempio, considera questa coda con un timeout di 5 minuti e le seguenti politiche di latenza dei giocatori:

1. Dedica 120 secondi alla ricerca di una posizione in cui tutte le latenze dei giocatori siano inferiori a 50 millisecondi.
2. Dedica 120 secondi alla ricerca di una posizione in cui tutte le latenze dei giocatori siano inferiori a 100 millisecondi.
3. Trascorri il tempo di coda rimanente fino al timeout cercando una posizione in cui tutte le latenze dei giocatori siano inferiori a 200 millisecondi.

Create queue

Queue settings

Name

The name must be unique and have 1-128 characters. Valid characters: A-Z, a-z, 0-9, and - (hyphen).

Timeout

Specify how long GameLift tries to place a game session before stopping.

 seconds

Must be 10-600 seconds.

 We recommend setting player latency policies, unless you're using GameLift FlexMatch.



Player latency policies - *optional*

Add policies to help place players into games with lower latency. Use multiple policies to reduce latency requirements per policy so that each player eventually finds a match.

0 seconds left to allocate 100%

Period start

Seconds

Period end

Seconds

Max player latency

Milliseconds

Remove

Seconds

Seconds

Milliseconds

Remove

Seconds

Seconds

Milliseconds

Remove

Add policy

Crea una coda per le istanze Spot

È possibile ottenere risparmi potenzialmente significativi sui costi di hosting utilizzando le flotte Spot. Per ulteriori dettagli sulle flotte Spot e su come utilizzarle, consulta [On-Demand Instances versus Spot Instances](#)

Se la tua soluzione di hosting di giochi include flotte Spot, devi utilizzare una coda per il posizionamento delle sessioni di gioco. Amazon GameLift Servers utilizza le code per cercare tra più risorse di hosting di giochi e selezionare quella migliore disponibile per ospitare una nuova sessione di gioco. Con le flotte Spot, le code sono particolarmente importanti per ridurre al minimo i costi di hosting ed evitare possibili interruzioni Spot. Questo argomento ti aiuta a configurare una coda resiliente che possa continuare a ospitare partite per i giocatori anche in caso di interruzioni, rallentamenti e interruzioni. Puoi personalizzare il modo in cui la coda assegna la priorità alle risorse di hosting disponibili in base a diversi fattori, tra cui i costi di hosting.

Utilizzi FlexMatch per l'abbinamento? Puoi utilizzare una coda con le flotte Spot per stabilire i piazzamenti delle sessioni di gioco per le tue partite.

Attività di implementazione per le flotte Spot

Quando crei o aggiorni la tua soluzione di hosting di giochi per utilizzare le flotte Spot, completa le seguenti attività. Per indicazioni più dettagliate su come creare una coda che ottimizzi la disponibilità e la resilienza di Spot, consulta [Riduci i costi di hosting dei giochi con le flotte Spot](#)

1. Scegli e crea una serie di destinazioni della flotta per la coda delle sessioni di gioco.

Inizia decidendo dove vuoi che la coda collochi le sessioni di gioco. Una coda può cercare tra più flotte per trovare il miglior posizionamento possibile. Ogni flotta ha un tipo di istanza ma può avere più posizioni geografiche. Le code con flotte che offrono varietà sia in termini di ubicazione che di tipo di istanza hanno maggiori probabilità di ottenere posizionamenti di successo. Consulta queste best practice per progettare una coda efficace e resiliente ottimizzata per SPOT.

2. Crea la tua coda di sessioni di gioco ottimizzata per SPOT.

Crea una coda e configurala per le tue flotte Spot. Consultare [Crea una coda per le sessioni di gioco](#) per assistenza con la creazione e la configurazione della nuova coda. È possibile utilizzare la Amazon GameLift Servers console o la AWS CLI per creare o modificare una coda.

- Aggiungi le destinazioni della flotta dalla Fase 1.
- Assegna la priorità all'ordine di destinazione in base alle esigenze. Per impostazione predefinita, Amazon GameLift Servers assegna la priorità in base al costo prima della destinazione, quindi l'ordine di destinazione viene utilizzato solo quando i costi più bassi tra le destinazioni sono uguali.

- Se desideri dare priorità ai costi di hosting dei giochi prima della latenza dei giocatori, fornisci una priorità di posizionamento personalizzata. Per informazioni, consulta [Dare la priorità al posizionamento delle sessioni di gioco](#).
3. Aggiorna gli altri componenti della soluzione per utilizzare la nuova coda.

Quando la soluzione utilizza una coda ottimizzata per SPOT per avviare nuove sessioni di gioco, la coda evita automaticamente di effettuare sessioni di gioco con flotte ad alta probabilità di interruzione. Cerca invece in tutte le flotte valide le risorse che corrispondono alle priorità definite, tra cui la latenza dei giocatori, i costi di hosting e l'ordine di destinazione.

- Se non lo utilizzi FlexMatch, aggiorna il servizio di backend per specificare la nuova coda ottimizzata per SPOT nelle richieste di sessione di gioco. Il servizio di backend invia richieste API per Amazon GameLift Servers conto del tuo client di gioco (utilizzando `StartGameSessionPlacement()`) e ogni richiesta deve specificare un nome di coda. Per assistenza con l'implementazione dei posizionamenti della sessione di gioco nel client di gioco, consultare [Crea sessioni di gioco](#).
 - Se lo utilizzi FlexMatch: aggiorna la configurazione del matchmaking per inviare le richieste di sessioni di gioco alla nuova coda ottimizzata per SPOT. Quando il sistema di matchmaking crea una partita tra giocatori, invia una richiesta di posizionamento della sessione di gioco alla coda designata per iniziare una nuova sessione di gioco per la partita. Solo le configurazioni di matchmaking con FlexMatch modalità impostata su «Gestita» possono designare una coda di posizionamento. Puoi aggiornare una configurazione di matchmaking utilizzando la AWS CLI o Amazon GameLift Servers la console ([vedi Modificare una configurazione di matchmaking](#)).
4. Controlla le prestazioni delle tue flotte e delle tue code Spot.

Visualizza le Amazon GameLift Servers metriche nella Amazon GameLift Servers console o con Amazon CloudWatch per esaminare le prestazioni. Per ulteriori informazioni sui parametri Amazon GameLift Servers, consulta [Monitora Amazon GameLift Servers con Amazon CloudWatch](#). I parametri chiave includono:

- Frequenza di interruzione: utilizza le `GameSessionInterruptions` metriche `InstanceInterruptions` and per tenere traccia del numero e della frequenza delle interruzioni relative a SPOT per istanze e sessioni di gioco. Le sessioni di gioco sulle istanze recuperate hanno uno stato e un motivo dello stato pari a `TERMINATED INTERRUPTED`
- Efficacia della coda: monitora le percentuali di successo del posizionamento, il tempo medio di attesa e la profondità della coda per verificare che le flotte Spot non influiscano sulle prestazioni in coda.

- Utilizzo della flotta: monitora i dati sulle istanze, sulle sessioni di gioco e sulle sessioni dei giocatori. L'utilizzo delle flotte On-Demand può essere un indicatore del fatto che le code impediscono il collocamento nelle flotte Spot per evitare interruzioni.

Best practices for queues with Spot fleets

Utilizza le seguenti best practice per creare flotte e code per le istanze Spot.

- Espandi la copertura geografica della coda. Anche se i tuoi giocatori sono raggruppati in un unico gruppo Regione AWS, aggiungi località adiacenti alla tua flotta Spot. Questo approccio migliora la capacità della coda di mantenere la capacità durante i rallentamenti regionali, le interruzioni e le interruzioni Spot. Le flotte con più sedi funzionano sia con istanze Spot che On-Demand.
- Diversificate la copertura dei tipi di istanze della coda. Amazon GameLift Servers valuta la fattibilità di Spot in base al tipo di istanza, quindi disporre di flotte Spot con una varietà di tipi di istanze riduce la possibilità che più flotte Spot non siano utilizzabili contemporaneamente. Includi almeno due flotte Spot con tipi di istanze diversi per ogni sede.

Note

I prezzi si basano sulle istanze utilizzate, non sul numero di flotte. Gestire cinque flotte con 10 istanze ciascuna equivale a gestire una flotta con 50 istanze a costi simili. I prezzi variano in base al tipo, alle dimensioni e all'ubicazione dell'istanza.

Suggerimenti per raggruppare i tipi di istanze Spot:

- Utilizza tipi di istanze della stessa famiglia, ad esempio `m6g.medium`, `m6g.large`, `m6g.xlarge`. I tipi di istanze più grandi costano di più, ma possono anche ospitare più sessioni di gioco contemporaneamente.
- Seleziona tipi di istanze ampiamente disponibili. In genere, le famiglie di vecchia generazione (come C5, M5 e R5) e le dimensioni comuni (come `.large`, `.xlarge` e `.2xlarge`) hanno una disponibilità migliore.
- Controlla la cronologia dei prezzi di 30-90 giorni nella console. Amazon GameLift Servers Cerca tipi di istanze con modelli di disponibilità coerenti.
- Usa la Amazon GameLift Servers console, lo strumento per la creazione del parco veicoli, per esplorare la copertura delle località, ad esempio i tipi di esempio.

- Aggiungi flotte On-Demand per aumentare la capacità di backup. L'hosting di giochi può passare alle flotte On-Demand ogni volta che le flotte Spot non sono disponibili. Metti almeno una flotta On-Demand in ogni sede per mantenere bassa la latenza dei giocatori. Aggiungi l'auto-scaling alle tue flotte di backup On-Demand, in modo da poterle mantenere ridimensionate fino a quando non saranno necessarie.
- Assegna alias a tutte le destinazioni della flotta. Crea alias per ciascuna destinazione della coda. Gli alias rendono più semplice ed efficiente ogni volta che è necessario sostituire le flotte.
- Applica una strategia di prioritizzazione delle code. Puoi personalizzare il modo in cui una coda assegna la priorità a dove collocare le sessioni di gioco (vedi per maggiori dettagli). [Dare la priorità al posizionamento delle sessioni di gioco](#) Per le code ottimizzate per Spot, l'assegnazione delle priorità in base al costo garantisce l'utilizzo di flotte Spot a basso costo ogni volta che è possibile.

Puoi anche dare priorità a determinate flotte specificando un ordine di destinazione. Ad esempio, alcuni utenti designano un set di flotte primarie per l'uso regolare e anche un set di flotte secondarie come backup. In questo scenario, imposta l'ordine di destinazione della coda in modo da elencare per prime le flotte primarie. Quindi configura l'ordine di priorità della coda con la destinazione seguita dal costo.

Personalizzazioni delle risorse di hosting

Questa sezione fornisce opzioni avanzate per la configurazione e la gestione Amazon GameLift Servers dell'infrastruttura per soddisfare requisiti specifici in termini di prestazioni, costi e operativi. In particolare, gli argomenti di questa sezione descrivono come personalizzare le risorse di hosting Amazon GameLift Servers gestito per adattarle al meglio al gioco e ai giocatori.

Alcune delle decisioni da prendere in considerazione:

- Dove distribuire le risorse di hosting per i tuoi giocatori? La latenza di gioco è un fattore importante nella selezione delle posizioni geografiche della flotta, ma ci sono altri fattori che variano in base alla località, tra cui il tipo di risorsa, la disponibilità e il costo.
- Quali tipi di EC2 istanze supporteranno al meglio il tuo gioco? Scegli tra i tipi di istanze disponibili che utilizzano la migliore combinazione di architettura di calcolo, memoria, archiviazione e capacità di rete.
- Di che dimensione del tipo di istanza hai bisogno? Scegli la dimensione del tipo di istanza in base ai requisiti di risorse (memoria e CPU) del software del server di gioco e ad altri fattori.

- La tua flotta dovrebbe utilizzare istanze On-Demand o Spot? Valuta se puoi trarre vantaggio dai prezzi Spot più bassi e se riduci Amazon GameLift Servers sufficientemente la possibilità che Spot interrompa le tue sessioni di gioco.
- Come vuoi che funzioni il software del tuo server di gioco su ogni istanza del parco istanze? La configurazione di runtime indica Amazon GameLift Servers quale software server eseguire e come.
- Per le flotte di container, la configurazione predefinita funziona per il tuo gioco? Amazon GameLift Servers svolge gran parte del lavoro per voi per ottimizzare le configurazioni della flotta di container, ma potete personalizzare la maggior parte delle impostazioni di configurazione.

Scegli le risorse di elaborazione per una flotta gestita

Per l'hosting Amazon GameLift Servers gestito, inclusi EC2 gestito e container gestiti, il servizio distribuisce i server di gioco su flotte di risorse informatiche in Cloud AWS. Quando crei una flotta gestita, desideri configurare le risorse di hosting per adattarle al meglio al tuo gioco. Questo argomento illustra i punti decisionali chiave nella scelta e nella configurazione delle flotte di hosting dei giochi.

Note

Se stai creando una soluzione ibrida con Anywhere e flotte Amazon GameLift Servers gestite, utilizza questi argomenti per progettare flotte gestite da integrare alle tue risorse autogestite. Per informazioni, consulta [Implementa flotte di hosting per Amazon GameLift Servers](#).

Argomenti

- [Posizioni geografiche](#)
- [Sistemi operativi](#)
- [Tipi di istanza](#)
- [On-Demand Instances versus Spot Instances](#)
- [Service Quotas](#)

Posizioni geografiche

Valuta dove intendi installare i tuoi server di gioco. In generale, vuoi mettere i server di gioco il più vicino possibile ai giocatori per offrire la migliore esperienza di gioco possibile. Per l'hosting Amazon GameLift Servers gestito, puoi scegliere di collocare i server di gioco in una qualsiasi delle Local

Zone supportate Regioni AWS e nelle Local Zone. Se stai creando una soluzione ibrida, considera in che modo l'implementazione gestita della flotta può integrare le sedi delle tue flotte Amazon GameLift Servers Anywhere autogestite.

Per la maggior parte degli scenari di sviluppo e test, l'implementazione in un'unica sede ha senso. Mentre ci si prepara al lancio e in seguito, ci sono molti motivi per implementare l'implementazione in più aree geografiche. Questi includono il supporto di un ampio gruppo di giocatori e il miglioramento della resilienza e dell'affidabilità complessive dell'hosting di giochi. Inoltre, più sedi possono migliorare l'esperienza dei giocatori, velocizzando il posizionamento delle sessioni di gioco e offrendo più scelte per ottimizzare i posizionamenti in termini di latenza e costi.

Per un elenco delle sedi supportate da Amazon GameLift Servers e ulteriori informazioni sulle sedi per tutti i tipi di flotta, consulta [Amazon GameLift Servers sedi di assistenza](#)

Flotte con più sedi

Una singola flotta gestita può distribuire risorse in più sedi. È possibile impostare manualmente la capacità per ogni singola sede in un parco veicoli con più sedi.

Vantaggi dell'utilizzo di una flotta con più sedi:

- Implementazione e gestione semplificate della flotta: fornisci il software del server di gioco e la configurazione del parco macchine e Amazon GameLift Servers li distribuisce sulle istanze del parco istanze in più sedi (crea una sola volta, distribuisce ovunque). In una flotta di produzione, puoi visualizzare e gestire tutte le sedi di una flotta anziché dover gestire più flotte, ognuna situata in una regione diversa.
- Disponibilità delle zone locali: se si desidera utilizzare una zona locale, è necessario creare un parco veicoli con più sedi con una Regione AWS sede principale e Local Zones come postazioni remote. Le Local Zones sono estensioni in Regioni AWS grado di offrire una latenza ancora inferiore alle aree e ai clienti che ne hanno bisogno. È possibile aggiungere una zona locale a qualsiasi parco macchine con più sedi; non è necessario includere la zona principale della zona locale. Regione AWS
- Compatibilità con le code per le sessioni di gioco: puoi creare code per il posizionamento delle sessioni di gioco con una o più flotte multiple. Questo approccio offre flessibilità alla coda nell'assegnazione delle priorità e nella scelta delle località in cui ospitare una nuova sessione di gioco.
- Utilizzo efficiente delle risorse: con l'auto-scaling attivato Amazon GameLift Servers, è possibile ottimizzare meglio la scalabilità della capacità in tutte le sedi di una flotta.

Suggerimenti per l'utilizzo di flotte con più sedi:

- Verifica le quote relative al numero di sedi per flotta. Regione AWS Vedere [Quote di servizio Amazon GameLift Servers](#).
- Non tutti i tipi di istanze sono disponibili in tutte le località. A seconda delle sedi scelte, potresti avere opzioni di tipo di istanza limitate. La Amazon GameLift Servers console fornisce strumenti utili per aiutarti a trovare il giusto equilibrio tra posizioni e tipi di istanze.
- Prendi in considerazione l'idea [UDP ping beacon](#) di utilizzarla per raccogliere i dati sulla latenza dei giocatori per tutte le sedi della tua flotta. Amazon GameLift Servers può utilizzare questi dati per ottimizzare le sessioni di gioco in modo da ridurre la latenza e impedire ai giocatori di partecipare a sessioni con una latenza inaccettabilmente elevata. Questi endpoint speciali accettano messaggi UDP anziché i tradizionali ping ICMP, fornendo misurazioni accurate della latenza per aiutare a selezionare le posizioni ottimali del parco.

Sistemi operativi

Tutte le istanze di una flotta gestita vengono distribuite con un'Amazon Machine Image (AMI) che fornisce un ambiente di runtime completo per il software del server di gioco. Per le flotte EC2 gestite, specifichi il sistema operativo della build del server di gioco quando carichi la build su. Amazon GameLift Servers Per le flotte di container gestite, specifichi il sistema operativo nella definizione del gruppo di contenitori. Per ulteriori informazioni sulle versioni AMI più recenti, vedere [Amazon GameLift Servers Versioni AMI](#).

Le versioni AMI vengono aggiornate regolarmente. Quando crei una nuova flotta, Amazon GameLift Servers assegna l'ultima versione disponibile dell'AMI che hai selezionato per la build del gioco. Tutte le istanze distribuite in quella flotta utilizzano la stessa versione. Per mantenere la versione dell'AMI aggiornata con gli ultimi aggiornamenti di sicurezza e software, devi sostituire regolarmente le tue flotte. Come best practice, consigliamo di sostituire le flotte gestite ogni 30 giorni per mantenere l'ambiente di runtime dei server di gioco. Per le linee guida, consulta [Best practice relative alla sicurezza di Amazon GameLift Servers](#).

Tipi di istanza

Il tipo di istanze di un parco istanze gestito determina il tipo di hardware da distribuire per tutte le istanze del parco istanze e i tipi di istanze sono generalmente disponibili in varie dimensioni. Tutte le flotte Amazon GameLift Servers gestite utilizzano istanze Amazon EC2 e supportano un'ampia gamma di tipi di istanze che offrono diverse combinazioni di potenza di calcolo, memoria, storage e funzionalità di rete. La disponibilità dei tipi di istanze varia a seconda delle posizioni scelte.

La Amazon GameLift Servers console fornisce strumenti utili per aiutarti a trovare i tipi di istanza giusti per la build del gioco e le posizioni di distribuzione. Per le flotte di container gestite, la console offre anche indicazioni sui requisiti di alimentazione e memoria della CPU del gioco.

Quando scegli tra i tipi di istanze disponibili per il tuo gioco, considera:

- L'architettura di calcolo del tuo server di gioco: x64 o Arm (AWS Graviton).

Note

Le istanze Graviton Arm richiedono una build del server per un'AMI Linux. Per C++ e C# è richiesto Server SDK 5.1.1 o versione successiva. Per Go è richiesto Server SDK 5.0 o versione successiva. Queste istanze non forniscono out-of-the-box supporto per l'installazione di Mono su Amazon Linux 2023 (AL2023) o Amazon Linux 2 (AL2).

- I requisiti di elaborazione, memoria e archiviazione della build del tuo server di gioco.
- La dimensione del tipo di istanza. Oltre a soddisfare i requisiti degli eseguibili del software del server di gioco, istanze di dimensioni maggiori possono eseguire più and/or contenitori di processi del server di gioco su ciascuna istanza. I fattori da considerare includono i costi (è più economico eseguire alcune istanze di grandi dimensioni o molte istanze piccole). Considera anche in che modo la capacità della sessione di gioco potrebbe essere influenzata dall'aggiunta o dalla rimozione di istanze durante gli eventi di ridimensionamento della flotta o quando si chiudono istanze non funzionanti. Se ogni istanza esegue più processi del server di gioco contemporaneamente, l'aggiunta o la rimozione di un'istanza può influire in modo significativo sulla capacità di hosting del gioco.

Per ulteriori informazioni su questi tipi di istanze, consulta [Tipi di istanze Amazon EC2](#).

On-Demand Instances versus Spot Instances

Le istanze On-Demand di Amazon EC2 e le istanze Spot offrono lo stesso hardware e le stesse prestazioni, ma differiscono in termini di disponibilità e costi.

Istanze on demand

Puoi acquistare un'istanza on demand quando ne hai bisogno e conservarla per tutto il tempo che desideri. Le istanze on demand hanno un costo fisso, il che significa che paghi solo per il tempo in cui le utilizzi. Non ci sono impegni a lungo termine.

Spot Instances

Le istanze Spot possono offrire un'alternativa conveniente alle istanze on demand utilizzando la capacità di elaborazione inutilizzata. AWS I prezzi delle istanze Spot variano in base alla domanda e all'offerta per ogni tipo di istanza in ogni sede. AWS può recuperare le istanze Spot con una notifica di due minuti ogni volta che ha bisogno di ripristinare la capacità e le sessioni di gioco in esecuzione attiva su un'istanza recuperata vengono interrotte.

Amazon GameLift Servers offre diversi strumenti per ridurre la probabilità di interruzioni delle sessioni di gioco tramite Spot. Un algoritmo di viabilità spot tiene traccia dei dati storici sul tipo di istanza per anticipare quando il rischio di interruzione raggiunge un punto critico ed evita di collocare nuove sessioni di gioco su istanze Spot di quel tipo. Se si verifica un'interruzione, i server di gioco possono utilizzare la notifica per terminare correttamente una sessione di gioco per i giocatori.

L'hosting di giochi con flotte Spot deve utilizzare una coda per il posizionamento delle sessioni di gioco. Una coda è in grado di dare priorità ai posizionamenti delle sessioni di gioco in base alla redditività della flotta Spot, ai costi e ad altri fattori. Consulta questi argomenti per ulteriori informazioni su come sfruttare Spot per l'hosting dei tuoi server di gioco:

- [Riduci i costi di hosting dei giochi con le flotte Spot](#)
- [Crea una coda per le istanze Spot](#)

Service Quotas

Puoi visualizzare le quote di servizio predefinite Amazon GameLift Servers e lo stato attuale delle quote Account AWS utilizzando i seguenti strumenti:

- Per informazioni generali sulle quote di servizio per Amazon GameLift Servers, consulta [Amazon GameLift Servers endpoint e quote](#) in. Riferimenti generali di AWS
- Per un elenco dei tipi di istanze disponibili per ubicazione per il tuo account, apri la pagina delle [quote di servizio](#) della console. Amazon GameLift Servers Questa pagina mostra anche l'uso corrente del tuo account per ogni tipo di istanza in ogni sede.
- Per un elenco delle quote correnti del tuo account, per esempio i tipi di istanza per regione, esegui il comando AWS Command Line Interface [describe-ec2-instance-limits](#) (AWS CLI). Questo comando restituisce il numero di istanze attive presenti nella regione predefinita (o in un'altra regione specificata).

[Mentre ti prepari a lanciare il gioco, compila un questionario di lancio nella console. Amazon GameLift Servers](#) Il Amazon GameLift Servers team utilizza il questionario di lancio per determinare le quote e i limiti corretti per il gioco.

Personalizza una flotta di Amazon GameLift Servers container

Gli argomenti di questa sezione descrivono alcune delle funzionalità opzionali per i contenitori Amazon GameLift Servers gestiti. Puoi scegliere di utilizzare una o tutte queste funzionalità.

Argomenti

- [Imposta i limiti delle risorse](#)
- [Comprendi l'allocazione della memoria della flotta di container](#)
- [Configurazione NVMe di Drive Access](#)
- [Designare contenitori essenziali](#)
- [Configurare le connessioni di rete](#)
- [Imposta i controlli sanitari per i container](#)
- [Imposta le dipendenze del container](#)
- [Configura una flotta di container](#)

Imposta i limiti delle risorse

Per ogni gruppo di contenitori, puoi determinare la quantità di memoria e potenza di calcolo di cui il gruppo di contenitori ha bisogno per eseguire il software. Amazon GameLift Servers si basa su queste informazioni per gestire le risorse all'interno del gruppo di contenitori. Inoltre, utilizza queste informazioni per calcolare quanti gruppi di container di server di gioco può contenere un'immagine della flotta. Puoi anche impostare limiti per singoli contenitori.

È possibile impostare un limite massimo di memoria e potenza di calcolo per un gruppo di contenitori. Per impostazione predefinita, queste risorse sono condivise da tutti i contenitori del gruppo. È possibile personalizzare ulteriormente la gestione delle risorse impostando limiti per i singoli contenitori.

Imposta limiti opzionali per i singoli contenitori

L'impostazione di limiti di risorse specifici per i contenitori consente di esercitare un maggiore controllo su come i singoli contenitori possono utilizzare le risorse del gruppo. Se non imposti limiti specifici del contenitore, tutti i contenitori del gruppo condividono le risorse del gruppo. La

condivisione offre una maggiore flessibilità nell'utilizzo delle risorse laddove sono necessarie. Inoltre, aumenta la possibilità che i processi competano tra loro e causino il fallimento dei container.

Imposta una delle seguenti `ContainerDefinition` proprietà per qualsiasi contenitore.

- `MemoryHardLimitMebibytes`— Imposta un limite massimo di memoria per il contenitore. Se il contenitore supera questo limite, si verifica un riavvio.
- `Vcpulimit`: riserva una quantità minima di risorse vCPU per l'uso esclusivo del contenitore. Il contenitore ha sempre a disposizione la quantità riservata. Può superare questo minimo in qualsiasi momento, se sono disponibili risorse aggiuntive. (1024 unità CPU equivalgono a 1 vCPU.)

Imposta i limiti totali delle risorse per un gruppo di contenitori

Se si impostano limiti per singoli contenitori, potrebbe essere necessario modificare la quantità di memoria e di risorse vCPU necessarie al gruppo di contenitori. L'obiettivo è allocare risorse sufficienti per ottimizzare le prestazioni del server di gioco. Amazon GameLift Servers utilizza questi limiti per calcolare come impacchettare i gruppi di container dei server di gioco su un'istanza di fleet. Li utilizzerai anche per scegliere un tipo di istanza per una flotta di container.

Calcola la memoria totale e la vCPU necessarie per un gruppo di contenitori. Considera i seguenti aspetti:

- Quali sono tutti i processi che vengono eseguiti su tutti i contenitori del gruppo di contenitori? Somma le risorse necessarie per questi processi. Prendi nota di eventuali limiti specifici del contenitore.
- Quanti processi simultanei dei server di gioco intendi eseguire in ciascun gruppo di container? Lo determini nell'immagine del contenitore del server di gioco.

In base alla stima dei requisiti del gruppo di contenitori, imposta le seguenti `ContainerGroupDefinition` proprietà:

- `TotalMemoryLimitMebibytes`— Imposta un limite massimo di memoria per il gruppo di contenitori. Tutti i contenitori del gruppo condividono la memoria allocata. Se si impostano limiti per singoli contenitori, il limite di memoria totale deve essere uguale o superiore al limite di memoria specifico del contenitore più elevato.
- `TotalVcpuLimit`— Imposta un limite massimo di vCPU per il gruppo di contenitori. Tutti i contenitori del gruppo condividono le risorse CPU allocate. Se imposti limiti dei singoli contenitori, il limite totale della CPU deve essere uguale o superiore alla somma di tutti i limiti

della CPU specifici del contenitore. Come procedura ottimale, valuta la possibilità di impostare questo valore per raddoppiare la somma dei limiti della CPU del contenitore.

Scenario di esempio

Supponiamo di definire un gruppo di contenitori di server di gioco con i seguenti tre contenitori:

- Il contenitore A è il nostro contenitore per server di gioco. Stimiamo i requisiti di risorse per un server di gioco a 512 MiB e 1024 CPU. Prevediamo che il contenitore esegua 1 processo server. Poiché questo contenitore esegue il nostro software più importante, non abbiamo impostato alcun limite di memoria o limite di riserva per vCPU.
- Il contenitore B è un contenitore di supporto con requisiti di risorse stimati in 1024 MiB e 1536 CPU. Abbiamo impostato un limite di memoria di 2048 MiB e un limite di riserva della CPU di 1024 CPU.
- Il contenitore C è un altro contenitore di supporto. Abbiamo impostato un limite di memoria rigida di 512 MiB e un limite di riserva della CPU di 512 CPU.

Utilizzando queste informazioni, impostiamo i seguenti limiti totali per il gruppo di contenitori:

- Limite di memoria totale: 7680 MiB. Questo valore supera il limite di memoria massimo (1024 MiB).
- Limite totale della CPU: 13312 CPU. Questo valore supera la somma del limite della CPU (1024+512 CPU).

Comprendi l'allocazione della memoria della flotta di container

Quando si Amazon GameLift Servers distribuiscono gruppi di container su un'istanza del parco istanze, non tutta la memoria dell'istanza è disponibile per i container. Amazon GameLift Servers riserva una parte della memoria dell'istanza per il sistema operativo, l'agente Amazon ECS e altri servizi di supporto. La quantità di memoria riservata varia in base alla memoria totale del tipo di istanza. La comprensione di questo sovraccarico consente di configurare le definizioni dei gruppi di contenitori per utilizzare appieno le risorse disponibili.

Formula di sovraccarico di memoria

Amazon GameLift Servers calcola la memoria disponibile per i gruppi di contenitori utilizzando i seguenti passaggi:

1. Determina la percentuale del buffer di memoria. Amazon GameLift Servers riserva una percentuale della memoria totale dell'istanza in base ai seguenti livelli:

Memoria di istanza (MiB)	Percentuale riservata
Meno di 5.000	8%
Da 5.000 a 9.999	6%
Da 10.000 a 89.999	5%
Da 90.000 a 199.999	4%
200.000 o più	3%

2. Calcola la memoria disponibile. Sottrai la memoria riservata dalla memoria totale dell'istanza:

```
AvailableMemory = InstanceMemory - round(InstanceMemory ×
BufferPercentage)
```

3. Sottrai la memoria del gruppo di contenitori per istanza. Se il tuo parco veicoli utilizza un gruppo di container per istanza, sottrai tale numero dalla memoria disponibile.
TotalMemoryLimitMebibytes Un gruppo di container per istanza viene eseguito su ogni istanza del parco veicoli.

```
AvailableMemory = AvailableMemory -
PerInstanceCGD.TotalMemoryLimitMebibytes
```

4. Account per il sovraccarico del router di log. Se la registrazione è abilitata per la flotta, Amazon GameLift Servers riserva altri 50 MiB per gruppo di container di server di gioco per il log router.
5. Calcola il numero massimo di gruppi di contenitori di server di gioco. Il numero massimo di gruppi di contenitori di server di gioco che rientrano nell'istanza in base alla memoria è:

```
MaxGroupsByMemory = floor(AvailableMemory /
(GameServerCGD.TotalMemoryLimitMebibytes + LogRouterMemory))
```

LogRouterMemory Dov'è 50 MiB se la registrazione è abilitata o 0 se la registrazione è disabilitata.

Note

La memoria è solo uno dei fattori che determinano il numero di gruppi di contenitori di server di gioco contenuti in un'istanza. Amazon GameLift Servers considera anche la capacità della vCPU e le porte di connessione disponibili e utilizza il minimo di tutti e tre i calcoli.

Esempio di calcolo della memoria

Prendi in considerazione una flotta che utilizza un'istanza `c5.xlarge` (8.192 MiB di memoria totale) con la registrazione abilitata:

1. La memoria dell'istanza è di 8.192 MiB, che rientra nel livello 5.000-9.999 (buffer al 6%)
2. Memoria riservata = rotonda ($8.192 \times 0,06$) = 492 MiB
3. Memoria disponibile = $8.192 - 492 = 7.700$ MiB
4. Se si utilizza un gruppo di contenitori per istanza con `TotalMemoryLimitMebibytes` 512: memoria disponibile = $7.700 - 512 = 7.188$ MiB
5. Se ogni gruppo di container di server `TotalMemoryLimitMebibytes` di gioco ha 1.024: $\text{MaxGroupsByMemory} = \text{floor}(7.188 / (1.024 + 50)) = \text{floor}(7.188 / 1.074) = 6$

Memoria disponibile per tipo di istanza

La tabella seguente mostra la memoria totale e la memoria disponibile (dopo il Amazon GameLift Servers buffer) per i tipi di istanza di uso comune. Utilizzate questi valori come punto di partenza per configurare le definizioni dei gruppi di contenitori. La colonna Memoria disponibile mostra la memoria disponibile per tutti i gruppi di contenitori sull'istanza, prima di sottrarre qualsiasi sovraccarico del gruppo di contenitori o del router di registro per istanza.

Tipo di istanza	Memoria totale (MiB)	Percentuale del buffer	Memoria disponibile (MiB)
<code>c5.large</code>	4,096	8%	3.768
<code>c5.xlarge</code>	8,192	6%	7.700
<code>c5.2xlarge</code>	16,384	5%	15.565

Tipo di istanza	Memoria totale (MiB)	Percentuale del buffer	Memoria disponibile (MiB)
c5.4xlarge	32,768	5%	31.130
c5.9xlarge	73,728	5%	70.042
c5.12xlarge	98.304	4%	94.372
c5.18xlarge	147.456	4%	141.558
c5.24xlarge	196.608	4%	188,744
m5.large	8,192	6%	7.700
m5.xlarge	16,384	5%	15.565
m5.2xlarge	32,768	5%	31.130
m5.4xlarge	65.536	5%	62.259
m5.8xlarge	131.072	4%	125.829
m5.12xlarge	196.608	4%	188,744
r5.large	16,384	5%	15.565
r5.xlarge	32,768	5%	31.130
r5.2xlarge	65.536	5%	62.259
r5.4xlarge	131.072	4%	125.829
c6i.large	4,096	8%	3.768
c6i.xlarge	8,192	6%	7.700
c6i.2xlarge	16,384	5%	15.565
c6i.4xlarge	32,768	5%	31.130
c6i.8xlarge	65.536	5%	62.259

Tipo di istanza	Memoria totale (MiB)	Percentuale del buffer	Memoria disponibile (MiB)
c7i.large	4,096	8%	3.768
c7i.xlarge	8,192	6%	7.700
c7i.2xlarge	16,384	5%	15.565
c7i.4xlarge	32,768	5%	31.130
c7i.8xlarge	65.536	5%	62.259
m7i.large	8,192	6%	7.700
m7i.xlarge	16,384	5%	15.565
m7i.2xlarge	32,768	5%	31.130
m7i.4xlarge	65.536	5%	62.259
m7i.8xlarge	131.072	4%	125.829
m7i.12xlarge	196.608	4%	188.744
r7i.large	16,384	5%	15.565
r7i.xlarge	32,768	5%	31.130
r7i.2xlarge	65.536	5%	62.259
r7i.4xlarge	131.072	4%	125.829

Per i tipi di esempio non elencati qui, puoi calcolare la memoria disponibile utilizzando la formula sopra descritta. Consulta la [documentazione sui tipi di istanze Amazon EC2](#) per la memoria totale del tipo di istanza scelto.

Configurazione NVMe di Drive Access

Nelle istanze di tipo d, l' NVMe unità viene montata automaticamente `/data` nella directory durante l'avvio dell'host. Per consentire ai contenitori di accedere allo storage SSD, imposta la seguente proprietà: `ContainerGroupDefinition MountPoints`

- `InstancePath`— Impostato su `/data` fare riferimento all' NVMe unità montata automaticamente sull'istanza host.
- `AccessLevel`— Scegliete il livello di accesso appropriato per le esigenze del contenitore (ad esempio, `READ_ONLY` o `READ_WRITE`).
- `ContainerPath`— (Facoltativo) Specificate il percorso in cui verrà montato il percorso dell'istanza all'interno del contenitore. Se non viene specificato, il valore predefinito è il percorso dell'istanza.

Per ulteriori informazioni sui punti di montaggio, consulta [ContainerMountPoint](#) Amazon GameLift Servers API Reference.

Designare contenitori essenziali

Per un gruppo di contenitori per istanza, designa ogni contenitore come essenziale o non essenziale. I gruppi di contenitori per istanza devono avere almeno un contenitore di supporto essenziale. Il contenitore essenziale svolge il lavoro fondamentale del gruppo di contenitori. Si prevede che il contenitore essenziale sia sempre in esecuzione. Se fallisce, l'intero gruppo di contenitori si riavvia.

Imposta la `ContainerDefinition` proprietà su `Essential` `true` o `false` per ogni contenitore.

Configurare le connessioni di rete

Puoi personalizzare l'accesso alla rete per consentire al traffico esterno di connettersi a qualsiasi container di una flotta di container. Ad esempio, devi stabilire connessioni di rete al contenitore che esegue i processi del server di gioco, in modo che i client di gioco possano unirsi e giocare al gioco. I client di gioco si connettono ai server di gioco utilizzando porte e indirizzi IP.

In una flotta di container, la connessione tra client e server non è diretta. Internamente, un processo in un container ascolta su una porta container. Esternamente, il traffico in entrata si connette a un'istanza della flotta utilizzando una porta di connessione. Amazon GameLift Servers mantiene le mappature tra le porte container interne e le porte di connessione rivolte verso l'esterno, in modo che il traffico in entrata venga indirizzato al processo corretto sull'istanza. Per recuperare le mappature correnti delle porte per uno specifico gruppo di contenitori, chiamate l'operazione.

[DescribeContainerGroupPortMappings](#) Per ulteriori informazioni sulla visualizzazione delle mappature delle porte, vedere. [Visualizza le mappature delle porte del contenitore](#)

Amazon GameLift Servers fornisce un ulteriore livello di controllo per le connessioni di rete. Ogni flotta di container dispone di un'impostazione delle autorizzazioni in entrata, che consente di controllare l'accesso a ciascuna porta di connessione rivolta verso l'esterno. Ad esempio, è possibile rimuovere le autorizzazioni per tutte le porte di connessione per impedire l'accesso ai container della flotta.

Puoi aggiornare le autorizzazioni in entrata, le porte di connessione e le porte container di una flotta.

Warning

Se fornisci un servizio personalizzato InstanceConnectionPortRange o InstanceInboundPermissions non Amazon GameLift Servers gestirai più nessuno dei due valori per la tua flotta. È necessario impostare entrambi i campi per evitare comportamenti indefiniti.

Imposta gli intervalli di porte del contenitore

Configura gli intervalli di porte del contenitore come parte della definizione di ogni contenitore. Questo è un parametro obbligatorio per la definizione di un gruppo di contenitori. È necessario configurare un numero sufficiente di porte per ospitare tutti i processi in esecuzione simultanea che richiedono l'accesso esterno. Alcuni contenitori non avranno bisogno di porte.

Il contenitore del server di gioco, che gestisce i server di gioco, necessita di una porta per ogni processo del server di gioco in esecuzione contemporaneamente. Il processo del server di gioco ascolta sulla porta assegnata e lo segnala a. Amazon GameLift Servers

Imposta gli intervalli delle porte di connessione

Configura la tua flotta di container con un set di porte di connessione. Le porte di connessione forniscono l'accesso esterno alle istanze del parco istanze che eseguono i container. Amazon GameLift Servers assegna le porte di connessione e le mappa alle porte dei container in base alle esigenze.

Per impostazione predefinita, Amazon GameLift Servers calcola il numero di porte necessarie per tutti i gruppi di container e imposta un intervallo di porte per soddisfarle. Ti consigliamo vivamente di utilizzare valori Amazon GameLift Servers calcolati, che vengono aggiornati quando distribuisce gli aggiornamenti a una definizione di gruppo di contenitori. Se è necessario personalizzare gli intervalli di porte di connessione, utilizzare le seguenti indicazioni.

Quando crei una flotta di container, definisci un intervallo di porte di connessione (vedi [ContainerFleet: InstanceConnectionPortRange](#)). Assicurati che l'intervallo contenga un numero sufficiente di porte da mappare a ogni porta container definita in tutti i container di entrambi i gruppi di container della flotta. Per calcolare le porte di connessione minime necessarie, utilizza la seguente formula:

```
[Total number of container ports defined for containers in the game server container group] * [Number of game server container groups per instance] + [Total number of container ports defined for containers in the per-instance container group]
```

È consigliabile raddoppiare il numero minimo di porte di connessione.

 Note

Il numero di porte di connessione può potenzialmente limitare il numero di gruppi di contenitori di server di gioco per istanza. Se una flotta dispone di un numero sufficiente di porte di connessione per un gruppo di container di server di gioco per istanza, Amazon GameLift Servers implementerà un solo gruppo di contenitori di server di gioco, anche se le istanze dispongono di una potenza di calcolo sufficiente per più gruppi di contenitori di server di gioco.

Imposta le autorizzazioni in entrata

Le autorizzazioni in entrata controllano l'accesso esterno a una flotta di container specificando quali porte di connessione aprire per il traffico in entrata. Puoi utilizzare questa impostazione per attivare e disattivare l'accesso alla rete di una flotta in base alle esigenze.

Per impostazione predefinita, Amazon GameLift Servers calcola il numero di porte necessarie per tutti i gruppi di container e imposta un intervallo di porte per soddisfarle. Ti consigliamo vivamente di utilizzare valori Amazon GameLift Servers calcolati, che vengono aggiornati quando distribuisce gli aggiornamenti a una definizione di gruppo di contenitori. Se è necessario personalizzare gli intervalli di porte di connessione, utilizzare le seguenti indicazioni.

Quando crei una flotta di container, definisci un set di autorizzazioni in entrata (vedi [ContainerFleet: InstanceInboundPermissions](#)). Le porte di autorizzazione in entrata devono corrispondere agli intervalli di porte di connessione della flotta.

 Note

Poiché le porte container vengono selezionate casualmente tra le `InstanceConnectionPortRange`, per garantire che le connessioni di sessione possano essere effettuate, tutte le porte in entrata `InstanceConnectionPortRange` devono essere coperte da porte in `InstanceInboundPermissions`

Scenario di esempio

Questo esempio illustra come impostare tutte e tre le proprietà della connessione di rete.

- Il gruppo di container di server di gioco della nostra flotta ha 1 contenitore, che esegue 1 processo di server di gioco.

Nella definizione del gruppo di contenitori del server di gioco, impostiamo il `PortConfiguration` parametro per questo contenitore come segue:

```
"PortConfiguration": {  
  "ContainerPortRanges": [ { "FromPort": 10, "ToPort": 20, "Protocol": "TCP"} ]  
}
```

- La nostra flotta dispone anche di un gruppo di container per istanza con 1 container. Ha 1 processo che richiede l'accesso alla rete. Nella definizione del contenitore per istanza, impostiamo il `PortConfiguration` parametro per questo contenitore come segue:

```
"PortConfiguration": {  
  "ContainerPortRanges": [ { "FromPort": 25, "ToPort": 25, "Protocol": "TCP"} ] }
```

- La nostra flotta è configurata con 20 gruppi di container di server di gioco per istanza di flotta. Alla luce di queste informazioni, possiamo usare la formula per calcolare il numero di porte di connessione di cui abbiamo bisogno:
 - Minimo: 21 porte [1 porta contenitore per server di gioco * 20 gruppi di contenitori di server di gioco per istanza + 1 porta container per istanza]
 - Migliore pratica: 42 porte [porte minime* 2]

Durante la creazione della flotta di container, impostiamo il `InstanceConnectionPortRange` parametro come segue:

```
"InstanceConnectionPortRange": { "FromPort": 1010, "ToPort": 1071 }
```

- Vogliamo consentire l'accesso a tutte le porte di connessione disponibili. Durante la creazione della flotta di container, impostiamo il `InstanceInboundPermissions` parametro come segue:

```
"InstanceInboundPermissions": [  
  {"FromPort": 1010, "ToPort": 1071, "IpRange": "10.24.34.0/23", "Protocol":  
    "TCP"} ]
```

Imposta i controlli sanitari per i container

Un contenitore si riavvia automaticamente se si verifica un errore del terminale e smette di funzionare. Se un contenitore è considerato essenziale, richiede il riavvio dell'intero gruppo di contenitori.

Tutti i contenitori del server di gioco vengono automaticamente considerati essenziali. I container di supporto possono essere considerati essenziali, ma devono disporre di un meccanismo per segnalare lo stato. Puoi impostare controlli di integrità anche per i contenitori di supporto non essenziali.

È possibile definire criteri personalizzati aggiuntivi per misurare lo stato dei contenitori e utilizzare un controllo dello stato per verificare tali criteri. Per impostare un controllo dello stato del contenitore, puoi definirlo in un'immagine del contenitore Docker o nella definizione del contenitore. Se imposti un controllo dello stato nella definizione del contenitore, questo sovrascrive qualsiasi impostazione nell'immagine del contenitore.

Imposta le seguenti `SupportContainerDefinition` proprietà per il controllo dello stato del contenitore:

- **Command**— Fornisci un comando che controlli alcuni aspetti dello stato del contenitore. Siete voi a decidere quali criteri utilizzare per misurare lo stato di salute. Il comando deve restituire un valore di uscita pari a 1 (non salutare) o 0 (sano).
- **StartPeriod**— Specificare un ritardo iniziale prima che gli errori del controllo sanitario inizino a contare. Questo ritardo dà al contenitore il tempo di avviare i propri processi.
- **Interval**— Decidi con che frequenza eseguire il comando health check. Con quale rapidità desiderate rilevare e risolvere un guasto del contenitore?

- **Timeout**— Decidi per quanto tempo attendere l'esito positivo o negativo prima di riprovare il comando health check. Quanto tempo deve impiegare per il completamento del comando di controllo dello stato di salute?
- **Retries**— Quante volte è necessario riprovare il comando health check prima di registrare un errore?

Imposta le dipendenze del container

All'interno di ogni gruppo di contenitori è possibile impostare le dipendenze tra i contenitori in base allo stato del contenitore. Una dipendenza influisce sul momento in cui il contenitore dipendente può avviarsi o chiudersi in base allo stato di un altro contenitore.

Un caso d'uso chiave per le dipendenze consiste nella creazione di sequenze di avvio e chiusura per il gruppo di contenitori.

Ad esempio, potresti volere che il contenitore A venga avviato per primo e venga completato correttamente prima dell'avvio dei contenitori B e C. A tale scopo, create innanzitutto una dipendenza per il contenitore B dal contenitore A, a condizione che il contenitore A venga completato correttamente. Quindi crea una dipendenza per il contenitore C sul contenitore A con la stessa condizione. Le sequenze di avvio si verificano in ordine inverso rispetto all'arresto.

Configura una flotta di container

Quando crei una flotta di container, prendi in considerazione i seguenti punti decisionali. La maggior parte di questi punti dipende dall'architettura e dalla configurazione del container.

Decidi dove vuoi distribuire la tua flotta

In generale, vuoi schierare le tue flotte geograficamente vicino ai giocatori per ridurre al minimo la latenza. Puoi distribuire la tua flotta di container su qualsiasi dispositivo che supporti. Regione AWS Amazon GameLift Servers Se desideri distribuire lo stesso server di gioco in altre località geografiche, puoi aggiungere località remote alla flotta, tra cui Regioni AWS Local Zones. Per una flotta con più sedi, puoi regolare la capacità in modo indipendente in ogni sede del parco veicoli. Per ulteriori informazioni sulle sedi del parco veicoli supportate, consulta [Amazon GameLift Servers sedi di assistenza](#).

Prendi in considerazione l'idea [UDP ping beacon](#) di raccogliere dati sulla latenza di rete in varie aree geografiche per anticipare la latenza tra i dispositivi dei giocatori e le potenziali sedi del parco veicoli. Questi endpoint speciali accettano messaggi UDP anziché i tradizionali ping ICMP,

fornendo misurazioni accurate della latenza per aiutare a selezionare le posizioni ottimali del parco.

Scegli un tipo e una dimensione di istanza per il tuo parco istanze

Amazon GameLift Servers supporta un'ampia gamma di tipi di istanze Amazon EC2, tutte disponibili per l'uso con una flotta di container. La disponibilità e il prezzo del tipo di istanza variano in base alla località. È possibile visualizzare un elenco dei tipi di istanze supportati, filtrati per posizione, nella Amazon GameLift Servers console (in Risorse, Istanze e quote di servizio).

Quando scegli un tipo di istanza, considera innanzitutto la famiglia di istanze. Le famiglie di istanze offrono varie combinazioni di CPU, memoria, archiviazione e funzionalità di rete. Ottieni maggiori informazioni sulle [famiglie di istanze EC2](#). All'interno di ogni famiglia hai una gamma di dimensioni di istanze tra cui scegliere. Considerate i seguenti aspetti quando selezionate la dimensione di un'istanza:

- Qual è la dimensione minima dell'istanza in grado di supportare il tuo carico di lavoro? Utilizza queste informazioni per eliminare i tipi di istanza troppo piccoli.
- Quali sono le dimensioni dei tipi di istanza più adatte alla tua architettura di container? Idealmente, dovresti scegliere una dimensione che possa ospitare più copie del gruppo di contenitori del tuo server di gioco con uno spreco minimo di spazio.
- Quale granularità di scalabilità è utile per il tuo gioco? La capacità di Scale Fleet prevede l'aggiunta o la rimozione di istanze e ogni istanza rappresenta la capacità di ospitare un numero specifico di sessioni di gioco. Considera quanta capacità desideri aggiungere o rimuovere con ogni istanza. Se la domanda dei giocatori varia di migliaia di minuti in minuto, potrebbe essere opportuno utilizzare istanze molto grandi in grado di ospitare centinaia o migliaia di sessioni di gioco. Al contrario, potresti preferire un controllo di ridimensionamento più preciso con tipi di istanze più piccoli.
- Sono disponibili risparmi sui costi in base alle dimensioni? Potresti scoprire che il costo di alcuni tipi di istanze varia in base alla località a causa della disponibilità.

Imposta altre impostazioni opzionali del parco veicoli

Puoi utilizzare le seguenti funzionalità opzionali durante la configurazione di una flotta di container:

- Configura i server di gioco per accedere ad altre AWS risorse. Per informazioni, consulta [Connect il server di gioco Amazon GameLift Servers ospitato ad altre AWS risorse](#).
- Proteggi le sessioni di gioco con giocatori attivi dalle interruzioni premature durante un evento a scala ridotta.

- Limita il numero di sessioni di gioco che un individuo può creare sulla flotta entro un periodo di tempo limitato.

Riduci i costi di hosting dei giochi con le flotte Spot

Quando si ospitano server di gioco multiplayer utilizzando l'hosting Amazon GameLift Servers gestito, le istanze Spot possono fornire un'alternativa conveniente alle istanze On-Demand. Il modello di prezzo Spot offre lo stesso hardware e le stesse prestazioni di On-Demand, ma con risparmi sui costi potenzialmente significativi (fino al 70-90%). Tuttavia, presentano un limite: quando è AWS necessario ripristinare la capacità, può recuperare queste istanze con una notifica di interruzione di due minuti.

Amazon GameLift Servers riduce il rischio di interruzioni per l'hosting di server di gioco. Amazon GameLift Servers prevede la probabilità di interruzioni sui tipi di istanze Spot ed evita di mettere a rischio le sessioni di gioco su eventuali istanze. Se si verifica un'interruzione rara, la notifica ti consente di terminare senza problemi una sessione di gioco per i giocatori.

Come Amazon GameLift Servers funziona con le flotte Spot

Quando configuri le flotte Spot per l'hosting di giochi, valuta Amazon GameLift Servers continuamente i tipi e le sedi delle istanze della flotta Spot per verificarne la fattibilità.

- L'algoritmo di fattibilità di Spot analizza i modelli di disponibilità recenti e i tassi di interruzione storici dei tipi di istanze Spot in base alla posizione.
- Sulla base di questa analisi, Amazon GameLift Servers identifica i tipi di istanze Spot e le posizioni in cui esiste un potenziale inaccettabile di interruzione della sessione di gioco. Intraprende le seguenti azioni:
 - Contrassegna la combinazione di tipo e posizione dell'istanza come temporaneamente non valida.
 - Quando si effettuano nuove sessioni di gioco, non viene presa in considerazione alcuna postazione della flotta Spot non utilizzabile. Di conseguenza, le sessioni di gioco vengono collocate solo nelle sedi della flotta Spot che hanno un'alta probabilità di ospitare server di gioco ininterrotti.
 - Prosciuga le istanze esistenti dal parco istanze Spot, anche se AWS non le recupera, quindi non pagherai per istanze che non possono essere utilizzate per l'hosting di giochi. Se la protezione delle sessioni di gioco è attivata, le istanze vengono chiuse solo dopo il completamento delle sessioni di gioco attive.

- Amazon GameLift Servers valuta continuamente i tipi di istanze e le ubicazioni della flotta Spot per garantire la fattibilità dell'hosting dei giochi. Quando un tipo di istanza precedentemente non utilizzabile diventa nuovamente utilizzabile sulla base di dati storici aggiornati, puoi ampliare nuovamente la flotta Spot e Amazon GameLift Servers riprendere a utilizzare tale istanza per le sessioni di gioco.

Considerazioni di natura progettuale

Quando progetti la tua soluzione per utilizzare le flotte Spot, considera i seguenti aspetti:

- Valuta la durata della sessione di gioco: la durata media delle sessioni di gioco può influire sull'efficacia di Spot per il gioco. Con sessioni di gioco più brevi, una maggiore rapidità di risposta consente alle sessioni di gioco di funzionare su tipi di istanze valide sulla base dei dati storici più recenti. Le sessioni di gioco più lunghe continuano a essere eseguite su tipi di istanze senza valutare i dati di fattibilità recenti, con un rischio maggiore di interruzione nel tempo.
- Valuta la disponibilità dei tipi di istanze: non tutte le sedi del parco istanze offrono tutti i tipi di istanze come Spot. Quando scegli un tipo di istanza per il tuo parco istanze Spot, utilizza lo strumento di creazione del parco istanze su Amazon GameLift Servers console per aiutarti a trovare i tipi di istanze Spot nelle località di cui hai bisogno. Con questo strumento, puoi selezionare le sedi del tuo parco veicoli e quindi visualizzare la disponibilità dei tipi di istanza in tali sedi.
- Crea flotte Spot con più sedi: puoi creare flotte Spot con più sedi. Un'unica flotta Spot multisede distribuisce istanze dello stesso tipo in più Local Region AWS Zones. L'algoritmo di fattibilità Spot valuta la fattibilità in base al tipo e alla posizione dell'istanza. Se l'ubicazione di una flotta Spot viene valutata come non praticabile, ciò non influisce sulle altre postazioni del parco veicoli, che possono comunque essere utilizzate per ospitare sessioni di gioco.
- Crea code con la diversità della flotta Spot: se utilizzi le flotte Spot per l'hosting dei giochi, devi impostare una coda per il posizionamento delle sessioni di gioco. Per ogni nuova richiesta di sessione di gioco, la coda cerca le risorse disponibili per l'hosting dei giochi e seleziona l'opzione migliore possibile. Con le flotte Spot, desideri una coda in grado di effettuare ricerche tra più flotte che variano in base alla località e al tipo di istanza e desideri includere almeno una flotta On-Demand come capacità di backup. Una coda multi-flotta ben progettata che offre diverse opzioni di posizionamento è altamente resiliente contro interruzioni, rallentamenti e interruzioni. Per ulteriori indicazioni sulla progettazione di una coda per Spot, consulta. [Crea una coda per le istanze Spot](#)
- Gestisci le interruzioni con garbo: configura il tuo server di gioco in modo da ridurre al minimo l'impatto sui giocatori in caso di interruzione di Spot. Quando AWS recupera un'istanza Spot, Amazon GameLift Servers trasmette la notifica di interruzione a tutti i processi del server interessati

utilizzando la funzione di callback dell'SDK del server. `onProcessTerminate()` Il gioco deve implementare questo callback per terminare correttamente la sessione di gioco. Per ulteriori informazioni, consulta [Rispondi a una notifica di chiusura del processo del server](#).

Note

AWS compie ogni sforzo per fornire la notifica prima di recuperare un'istanza, ma è possibile che AWS recuperi l'istanza Spot prima che arrivi l'avviso. Dovresti anche preparare il tuo server di gioco per gestire interruzioni impreviste.

- Configura l'auto-scaling per le tue flotte di backup per mantenere il servizio durante un'interruzione Spot. L'auto-scaling con tracciamento del target mantiene un buffer di capacità e si ridimensiona automaticamente in base alla domanda. Con l'auto-scaling, le flotte di backup (Spot o On-Demand) inizieranno ad aumentare la capacità ogni volta che iniziano a ricevere più richieste di sessioni di gioco.

Per sostituire rapidamente la capacità persa quando una flotta Spot diventa inutilizzabile, un meccanismo di scalabilità personalizzato può utilizzare le metriche disponibili relative alla coda e alla flotta per avviare una rapida scalabilità delle flotte di backup. Rileva quando le flotte Spot diventano inutilizzabili con metriche come, e. `FirstChoiceOutOfCapacity` `FirstChoiceNotViable` `PercentAvailableGameSessions` Stima le esigenze di capacità sostitutiva analizzando i dati metrici recenti. `PlacementsStarted` Dopo aver ridimensionato le flotte di backup per gestire la domanda immediata, può prendere il sopravvento l'auto-scaling normale.

- Integrazione con FlexMatch: se la tua soluzione utilizza un FlexMatch matchmaker, non ci sono requisiti speciali per le flotte Spot. Puoi configurare un matchmaker per utilizzare una coda con le flotte Spot. Amazon GameLift Servers assegna automaticamente la priorità ai piazzamenti delle partite tra le flotte Spot e On-Demand, anche quando si effettuano nuove sessioni di gioco e quando si riempiono gli slot vuoti nelle sessioni di gioco esistenti.

Ottimizza la configurazione del runtime del server di gioco su server gestiti Amazon GameLift Servers

Puoi configurare la configurazione di runtime di una flotta EC2 gestita per eseguire più processi del server di gioco per istanza. Questo utilizza le risorse di hosting in modo più efficiente.

In che modo un parco istanze gestisce più processi

Amazon GameLift Servers utilizza la configurazione di runtime di una flotta per determinare il tipo e il numero di processi da eseguire su ciascuna istanza. Una configurazione di runtime contiene almeno una configurazione del processo del server che rappresenta un eseguibile del server di gioco. Puoi definire configurazioni aggiuntive dei processi del server per eseguire altri tipi di processi relativi al tuo gioco. Ogni configurazione di processo server contiene le seguenti informazioni:

- Il nome e il percorso del file eseguibile della build di gioco.
- Parametri (facoltativi) da passare al processo all'avvio.
- Il numero di processi da eseguire contemporaneamente.

Quando un'istanza del parco istanze si attiva, avvia il set di processi del server definiti nella configurazione di runtime. Con più processi, l'avvio di ogni processo viene Amazon GameLift Servers scaglionato. I processi server hanno una durata limitata. Al termine, Amazon GameLift Servers avvia nuovi processi per mantenere il numero e il tipo di processi del server definiti nella configurazione di runtime.

Puoi modificare una configurazione di runtime in qualsiasi momento aggiungendo, modificando o rimuovendo le configurazioni del processo del server. Ogni istanza verifica regolarmente la presenza di aggiornamenti alla configurazione di runtime del parco macchine per implementare le modifiche. Ecco come Amazon GameLift Servers adotta le modifiche alla configurazione di runtime:

1. L'istanza invia una richiesta Amazon GameLift Servers per la versione più recente della configurazione di runtime.
2. L'istanza confronta i processi attivi con l'ultima configurazione di runtime e quindi esegue le seguenti operazioni:
 - Se la configurazione di runtime aggiornata rimuove un tipo di processo del server, i processi server attivi di questo tipo continuano a funzionare fino al termine. L'istanza non sostituisce questi processi del server.
 - Se la configurazione di runtime aggiornata riduce il numero di processi simultanei per un tipo di processo server, i processi server in eccesso di questo tipo continuano a funzionare fino al termine. L'istanza non sostituisce questi processi server in eccesso.
 - Se la configurazione di runtime aggiornata aggiunge un nuovo tipo di processo del server o aumenta i processi concorrenti per un tipo esistente, l'istanza avvia nuovi processi del server, fino al Amazon GameLift Servers massimo. In questo caso, l'istanza avvia nuovi processi del server al termine dei processi esistenti.

Ottimizza una flotta per più processi

Per utilizzare più processi su una flotta, procedi come segue:

- [Crea una build](#) che contenga gli eseguibili del server di gioco che desideri distribuire in una flotta, quindi carica la build su Amazon GameLift Servers. Tutti i server di gioco di una build devono funzionare sulla stessa piattaforma e utilizzare l'SDK del server per Amazon GameLift Servers.
- Creare una configurazione di runtime con una o più configurazioni di processo del server e più processi simultanei.
- Integra i client di gioco con la versione AWS SDK 2016-08-04 o successiva.

Per ottimizzare le prestazioni della flotta, ti consigliamo di fare quanto segue:

- Gestisci gli scenari di arresto dei processi del server in modo da Amazon GameLift Servers poter riciclare i processi in modo efficiente. Esempio:
 - Aggiungi una procedura di spegnimento al codice del server di gioco che richiami l'API del server. `ProcessEnding()`
 - Implementa la funzione di callback `OnProcessTerminate()` nel codice del server di gioco per gestire le richieste di terminazione da Amazon GameLift Servers.
- Assicurati che ciò Amazon GameLift Servers chiuda e riavvii i processi del server non integri. Riporta lo stato di salute a Amazon GameLift Servers implementando la funzione di `OnHealthCheck()` callback nel codice del server di gioco. Amazon GameLift Servers chiude automaticamente i processi del server segnalati come non integri per tre segnalazioni consecutive. Se non si esegue l'implementazione `OnHealthCheck()`, si Amazon GameLift Servers presume che un processo server sia integro, a meno che il processo non risponda a una comunicazione.

Scegli il numero di processi per istanza

Quando decidi il numero di processi simultanei da eseguire su un'istanza, tieni presente quanto segue:

- Amazon GameLift Servers limita ogni istanza a un [numero massimo di processi simultanei](#). La somma di tutti i processi simultanei per le configurazioni dei processi del server di una flotta non può superare questa quota.
- Per mantenere livelli di prestazioni accettabili, il tipo di istanza Amazon EC2 potrebbe limitare il numero di processi che possono essere eseguiti contemporaneamente. Prova diverse

configurazioni per il tuo gioco per trovare il numero giusto di processi per il tuo tipo di istanza preferito.

- Amazon GameLift Servers non esegue più processi simultanei rispetto al numero totale configurato. Ciò significa che la transizione dalla precedente configurazione di runtime alla nuova configurazione potrebbe avvenire gradualmente.

Collabora con l'Amazon GameLift Servers agente

L'Amazon GameLift Servers agente supervisiona l'esecuzione dei processi dei server di gioco sulle vostre Amazon GameLift Servers flotte. L'agente viene distribuito su ogni computer di una flotta e fornisce la gestione automatizzata dei processi, la gestione dell'hosting e la registrazione per l'elaborazione. Per utilizzare l'Agente, è necessario che la build del server di gioco sia integrata con l'SDK del server per Amazon GameLift Servers la versione 5.x o successiva.

L'Amazon GameLift Servers agente è disponibile esternamente per l'uso con Amazon GameLift Servers flotte che non sono flotte gestite. EC2 (EC2 Le flotte gestite gestiscono automaticamente le attività dell'agente.) Puoi scegliere di gestire Amazon GameLift Servers flotte, incluse le flotte Anywhere, con o senza l'agente. Senza l'agente, è necessario fornire una soluzione alternativa per il completamento delle attività richieste.

Quando viene distribuito su un computer, l'Amazon GameLift Servers agente deve essere avviato prima dell'avvio di qualsiasi processo del server di gioco. All'avvio, l'agente completa le seguenti attività:

- Registra l'elaborazione con una flotta Amazon GameLift Servers Anywhere utilizzando l'API. [RegisterCompute](#)
- Richiama l'[GetComputeAuthToken](#) API per recuperare un token di autorizzazione e lo archivia per l'utilizzo da parte dei processi del server in esecuzione sul computer.
- Imposta la variabile di ambiente WebSocket URL per il calcolo e stabilisce una WebSocket connessione al servizio. Amazon GameLift Servers
- Richiede al servizio la versione più recente della configurazione di runtime della Amazon GameLift Servers flotta.
- Avvia e arresta i processi del server in base alle istruzioni di configurazione del runtime.

Il codice sorgente e le istruzioni di compilazione per l'Amazon GameLift Servers agente sono disponibili nell'[Amazon GameLift Servers agente](#) GitHub.

Informazioni sull'agente

L'Amazon GameLift Servers agente è progettato per gestire le seguenti attività per le vostre flotte:

Gestione dei processi

- Avvia nuovi processi del server come definito nelle istruzioni di runtime. L'agente potrebbe utilizzare una configurazione di runtime personalizzata distribuita con l'agente. In alternativa, puoi fornire una `RuntimeConfiguration` come parte della definizione del tuo parco veicoli. Questo approccio ha il vantaggio di poter modificare la configurazione di runtime della flotta in qualsiasi momento. L'agente richiede periodicamente configurazioni di runtime aggiornate dal Amazon GameLift Servers servizio.
- Monitora le attivazioni dei processi del server e termina i processi quando non si attivano in tempo.
- Invia battiti cardiaci a. Amazon GameLift Servers Se l'agente non riesce a inviare battiti cardiaci, il computer potrebbe essere contrassegnato come obsoleto.
- Indica Amazon GameLift Servers quando termina un processo del server. Amazon GameLift Servers utilizza queste informazioni per monitorare la disponibilità del server di gioco per il posizionamento delle sessioni di gioco.
- Emette eventi di flotta per i processi del server, tra cui:
 - `SERVER_PROCESS_INVALID_PATH`: I parametri di avvio del processo del server di gioco non erano configurati correttamente.
 - `SERVER_PROCESS_TERMINATED_UNHEALTHY`: Il processo del server di gioco non ha segnalato un controllo sanitario valido entro 3 minuti dall'attivazione ed è stato quindi interrotto.
 - `SERVER_PROCESS_FORCE_TERMINATED`: il processo del server di gioco non è terminato correttamente dopo l'`OnProcessTerminate()` invio entro 30 secondi.
 - `SERVER_PROCESS_CRASHED`: Un processo del server di gioco si è bloccato per qualche motivo.

Gestione del calcolo

- Riceve messaggi dal Amazon GameLift Servers servizio per arrestare l'elaborazione.
- Richiede l'interruzione del calcolo da. Amazon GameLift Servers

Registrazione di log

- Carica i log in un bucket Amazon S3 nel tuo account. AWS

Designazione astratta e Amazon GameLift Servers della flotta con un alias

Un Amazon GameLift Servers alias viene utilizzato per astrarre una destinazione di hosting. Le destinazioni di hosting indicano Amazon GameLift Servers dove cercare le risorse disponibili per ospitare una nuova sessione di gioco per i giocatori. Gli alias sono utili nei seguenti scenari:

- Se il gioco non utilizza code con più flotte per il posizionamento delle sessioni di gioco, richiede nuove sessioni di gioco specificando un ID flotta. Amazon GameLift Servers Nel corso di una partita, sostituirai la flotta più volte, per aggiornare una build del server, aggiornare l'hardware e il sistema operativo di hosting o risolvere problemi di prestazioni. Usa un alias per estrarre l'ID della flotta in modo da poter trasferire senza problemi il traffico dei giocatori da un parco giocatori esistente a quello nuovo.
- Se vuoi fare qualcosa di diverso, basta creare una nuova sessione di gioco quando un client di gioco ne richiede una. Ad esempio, potresti voler indirizzare i giocatori che utilizzano un out-of-date client verso un sito Web di aggiornamento.

Un alias deve specificare una strategia di routing. Esistono due tipi. Una semplice strategia di routing indirizza il traffico dei giocatori verso un ID flotta specifico, che puoi aggiornare per reindirizzare il traffico. Una strategia di routing del terminale invia un messaggio al client invece di creare una nuova sessione di gioco. Puoi modificare la strategia di routing di un alias in qualsiasi momento.

Se utilizzi una coda per il posizionamento delle sessioni di gioco, non hai bisogno di un alias per reindirizzare il traffico quando sostituisci una flotta. Con una coda, puoi semplicemente aggiungere la nuova flotta e rimuovere quella vecchia. Questa azione non è visibile ai giocatori, perché le nuove richieste di sessioni di gioco vengono soddisfatte automaticamente utilizzando la nuova flotta. Non ha alcun impatto sulle sessioni di gioco esistenti. Puoi identificare le destinazioni delle code utilizzando un ID flotta o un alias.

Argomenti

- [Crea un Amazon GameLift Servers alias](#)

Crea un Amazon GameLift Servers alias

Questo argomento descrive come creare un Amazon GameLift Servers alias da utilizzare per il posizionamento delle sessioni di gioco.

Per creare un alias

Usa uno dei Amazon GameLift Servers console o AWS Command Line Interface (AWS CLI) per creare un alias.

Console

Nella [Amazon GameLift Servers console](#), usa il pannello di navigazione per aprire la pagina Alias.

1. Scegli Crea alias.
2. Inserisci un nome alias. Ti consigliamo di includere caratteristiche significative nel nome dell'alias per facilitare la visualizzazione di un elenco di alias.
3. Inserisci una descrizione dell'alias, se necessario.
4. Scegliete una strategia di routing per l'alias.
 - a. Se scegli una strategia di routing semplice, seleziona un ID flotta dall'elenco da associare a questo alias. L'elenco include tutte le flotte attualmente selezionate. Regione AWS È necessario creare un alias nella stessa regione della flotta.
 - b. Se scegli una strategia di routing del terminale, inserisci il valore di stringa che desideri Amazon GameLift Servers per tornare a un client di gioco in risposta a una richiesta di sessione di gioco. Una richiesta con un alias di terminale genera un'eccezione con il messaggio incorporato.
5. (Facoltativo) Aggiungi tag alla risorsa alias. Ogni tag è composto da una chiave e da un valore opzionale, entrambi personalizzabili. Assegna tag alle AWS risorse che desideri classificare in modi utili, ad esempio per scopo, proprietario o ambiente. Scegli Aggiungi nuovo tag per ogni tag che desideri aggiungere.
6. Quando sei pronto per implementare la nuova flotta, scegli Crea.

AWS CLI

Usa il [create-alias](#) comando per creare un alias. Amazon GameLift Servers crea la risorsa alias con l'impostazione predefinita corrente Regione AWS (oppure puoi aggiungere un tag `--region` per specificarne uno diverso). Regione AWS

Come minimo, includi un nome alias e una strategia di routing. Per una strategia di routing semplice, specifica l'ID di una flotta nella stessa regione dell'alias. Per una strategia di routing dei terminali, fornisci una stringa di messaggi.

Gestione di una soluzione di Amazon GameLift Servers hosting (lancio e oltre)

Questa sezione fornisce indicazioni su come gestire le Amazon GameLift Servers risorse a lungo termine. Avviare con successo la tua partita multigiocatore è solo l'inizio. La gestione di una soluzione di hosting di produzione richiede un'attenzione costante alle prestazioni, all'ottimizzazione dei costi, agli aggiornamenti di sicurezza e all'esperienza dei giocatori. Questa sezione fornisce indicazioni sugli aspetti operativi dell'utilizzo Amazon GameLift Servers in produzione, dalle attività di lancio sin dal primo giorno alla manutenzione e all'ottimizzazione a lungo termine.

Che tu stia preparando il lancio, gestendo una partita dal vivo o ottimizzando una soluzione di hosting consolidata, questi argomenti ti aiuteranno a mantenere un hosting di gioco affidabile ed economico, scalabile in base alla tua base di giocatori.

Gestisci le risorse di hosting di giochi con Amazon GameLift Servers

Gestisci Amazon GameLift Servers le risorse, tra cui build, flotte, code delle sessioni di gioco e matchmakers. Puoi utilizzare la Amazon GameLift Servers console per scansionare gli elenchi di risorse o visualizzare i dettagli delle risorse, comprese le metriche. Puoi anche recuperare le informazioni sulle risorse utilizzando l' AWS SDK o la AWS CLI per Amazon GameLift Servers

Amazon GameLift Servers funzionalità della console

Usa la Amazon GameLift Servers console per visualizzare e gestire le risorse di hosting dei giochi e le attività di hosting in corso quasi in tempo reale. La console offre un'interfaccia grafica per la maggior parte delle funzionalità dell'API di servizio per Amazon GameLift Servers. È possibile utilizzare la console per:

- Usa la dashboard per un'istantanea di alto livello. Puoi vedere i numeri e lo stato attuale di tutte le tue risorse di Amazon GameLift Servers hosting e seguire i link per ottenere dettagli sulle singole risorse.
- Gestisci le singole risorse di hosting. Puoi creare, visualizzare ed eliminare tutte le Amazon GameLift Servers risorse e aggiornarne le proprietà mutabili. Puoi anche visualizzare determinati tipi di attività di hosting, come gli eventi e le metriche delle prestazioni.

- Interagisci con le attività di gioco e di sessione dei giocatori. Puoi tenere traccia delle sessioni di gioco e delle sessioni dei giocatori per flotta e utilizzare queste informazioni per risolvere i problemi relativi alle sessioni di gioco. Visualizza i dettagli sulle sessioni di gioco, visualizza le sessioni dei giocatori per ogni sessione di gioco e cerca l'attività dei giocatori in più sessioni di gioco. Se necessario, puoi anche chiudere singole sessioni di gioco.

Argomenti

- [Dashboard di hosting nella Amazon GameLift Servers console](#)
- [build di server di gioco](#)
- [Flotte nella console Amazon GameLift Servers](#)
- [Alias nella console Amazon GameLift Servers](#)
- [Code delle sessioni di gioco nella console Amazon GameLift Servers](#)

Dashboard di hosting nella Amazon GameLift Servers console

Usa la dashboard della Amazon GameLift Servers console per ottenere un'istantanea di alto livello sullo stato attuale delle risorse di Amazon GameLift Servers hosting nel tuo account. AWS La Amazon GameLift Servers dashboard offre una visualizzazione di quanto segue:

- Il numero di build negli stati Ready, Initialized e Failed. Scegli Visualizza le build per i dettagli sulle build nella tua regione attuale.
- Il numero di flotte in tutti gli stati. Scegli Visualizza flotte per informazioni dettagliate sulle flotte nella tua regione attuale.
- Le tue risorse attuali.
- Annunci di nuove funzionalità e servizi.

Per aprire il pannello di controllo Amazon GameLift Servers

- Nella [Amazon GameLift Servers console](#), nel riquadro di navigazione, scegli Dashboard.

Dalla dashboard puoi:

- Prepara il gioco per il lancio selezionando Prepara al lancio e compilando il questionario di lancio corrispondente.

- Richiedi aumenti delle quote di servizio in preparazione al lancio o in risposta ai lanci scegliendo [Visualizza le quote di servizio](#).
- Visualizza i post del blog e le informazioni dettagliate sulle nuove funzionalità selezionando il link nella sezione Funzionalità.

GameLift > Dashboard

Dashboard

Build status overview

Viewing data for all builds in N. Virginia region.

[View builds](#)

✔ Ready
1

⊖ Initialized
0

✖ Failed
0

Fleet status overview

Viewing data for all fleets in N. Virginia region.

[View fleets](#)

✔ Active
0

⊖ New
0

⊖ Deleting
0

✖ Error
0

⊖ In progress
0

⊖ Terminated
0

Resources (1)

[View service quotas](#)

Resource type	Count
Builds	1
Scripts	0
Fleets	0
Aliases	0
Queues	0
Matchmaking rule sets	0
Matchmaking configurations	0

Prepare for your game launch

[Learn more](#)

Fill out a launch questionnaire

Fill out our game launch questionnaire and email it to the GameLift launch team to ensure a smooth launch. The GameLift launch team will verify your GameLift setup and service limits, preparing you for launch.

[Prepare to launch](#)

Features spotlight

Updates on features available in N. Virginia region

March 22, 2022

Updates to Amazon GameLift FlexMatch for greater flexibility

October 28, 2021

New Asia Pacific (Osaka) region and Graviton2 support for Amazon GameLift

build di server di gioco

La risorsa di compilazione rappresenta il software del server di gioco. Carichi il pacchetto di build su Amazon GameLift Servers per l'implementazione su flotte EC2 gestite.

Visualizza le informazioni sulle build dei server di gioco nella Amazon GameLift Servers console o utilizzando l'SDK o AWS per. Amazon GameLift Servers

Console

Nella pagina Builds della [Amazon GameLift Serversconsole](#), puoi visualizzare e gestire tutte le build di server di gioco che hai caricato Amazon GameLift Servers per la distribuzione su flotte EC2 gestite. Nel pannello di navigazione, scegli Hosting, Managed EC2, Builds.

La pagina Builds mostra le seguenti informazioni per ogni build. Puoi modificare il contenuto della tabella secondo necessità utilizzando lo strumento Preferenze (vedi



icona

nell'angolo in alto a destra della tabella). Le preferenze personalizzate vengono salvate nell'utente AWS del tuo account e vengono applicate automaticamente ogni volta che visualizzi questa pagina.

Note

La pagina Builds mostra solo le build nella AWS regione corrente.

- Nome: il nome associato alla build caricata.
- Stato: lo stato della build. Visualizza uno dei tre messaggi di stato:
 - Inizializzato: il caricamento non è iniziato o è ancora in corso.
 - Pronto: la build è pronta per la creazione della flotta.
 - Fallita: la compilazione è scaduta prima della Amazon GameLift Servers ricezione dei file binari.
- Ora di creazione: la data e l'ora in cui hai caricato la build. Amazon GameLift Servers
- ID build: l'ID univoco assegnato alla build al momento del caricamento.
- Versione: l'etichetta della versione associata alla build caricata.
- Sistema operativo: il sistema operativo su cui viene eseguita la build. Il sistema operativo di build determina il sistema operativo da Amazon GameLift Servers installare sulle istanze di un parco istanze.
- Dimensione: la dimensione, in megabyte (MB), del file di build caricato su. Amazon GameLift Servers

- Flotte: il numero di flotte dispiegate durante la build.

Da questa pagina è possibile eseguire queste operazioni:

- Visualizzare i dettagli delle build. Scegli il nome di una build per aprire la relativa pagina dei dettagli della build.
- Creare un nuovo parco istanze da una build. Seleziona una build, quindi scegli Crea flotta.
- Filtrare e ordinare l'elenco delle build. Utilizzare i controlli nella parte superiore della tabella.
- Eliminare una build. Seleziona una build, quindi scegli Elimina.

Dettagli della build

Nella pagina Builds, scegli il nome di una build per aprirne la pagina dei dettagli. La sezione Panoramica della pagina dei dettagli mostra le stesse informazioni di riepilogo della build della pagina Builds. [La sezione Flotte mostra un elenco di flotte su cui è in esecuzione la build, incluse le stesse informazioni di riepilogo della pagina Flotte.](#)

AWS SDK

Utilizza i seguenti comandi AWS CLI per recuperare informazioni su questa risorsa:

- [ListBuild](#)
- [DescribeBuild](#)

Aggiorna una build del server di gioco

Quando distribuisce la build del tuo server di gioco per l'hosting EC2 Amazon GameLift Servers gestito, carichi il software del server di gioco e crei una Amazon GameLift Servers risorsa di build. Dopo aver creato una Amazon GameLift Servers build, puoi aggiornare i metadati della build, ma non puoi aggiornare i file di build stessi. Per distribuire gli aggiornamenti alla build del server di gioco, devi caricare un nuovo set di file e creare una nuova Amazon GameLift Servers build utilizzando il AWS CLI [upload-build](#) comando. In alternativa, puoi usare il [create-build](#) comando per caricare una nuova build da un bucket Amazon S3 che controlli. Quindi distribuisce la nuova build creando una nuova flotta per essa.

Puoi aggiornare i metadati di una build, inclusi il nome e la descrizione. Per queste attività, usa la Amazon GameLift Servers console o il [update-build](#) AWS CLI comando.

Automatizza gli aggiornamenti della build del gioco

Segui questi suggerimenti per automatizzare e semplificare il processo di aggiornamento delle build dei server di gioco per le flotte gestite: Amazon GameLift Servers

- Usa le code delle sessioni di gioco e sostituisci le flotte secondo necessità. Quando invii le richieste di sessioni di gioco a Amazon GameLift Servers, specifica una coda per le sessioni di gioco anziché una flotta specifica. Con le code, puoi aggiungere flotte con una nuova build e rimuovere quelle vecchie se necessario. Per ulteriori informazioni, consulta [Configura il posizionamento delle sessioni di gioco](#).
- Usa gli alias per trasferire i giocatori a una nuova build di gioco. Quando invii richieste di sessioni di gioco a Amazon GameLift Servers, specifica un alias della flotta anziché un ID della flotta. Per ulteriori informazioni, consulta [Crea un Amazon GameLift Servers alias](#).
- Configurato per lo sviluppo iterativo. Durante lo sviluppo del gioco, esplora le opzioni per configurare un ambiente di test ospitato che supporti lo sviluppo iterativo rapido. Per informazioni, consulta [Configurazione per lo sviluppo iterativo con Anywhere Amazon GameLift Servers](#).

Prova queste risorse del [Amazon GameLift Servers Toolkit](#) su: GitHub

Strumento di aggiornamento rapido (solo per lo sviluppo)

Questo strumento ti aiuta a modificare le build dei server di gioco che sono già distribuite sui computer di una flotta EC2 gestita, facendoti risparmiare tempo durante le iterazioni di sviluppo rapide. Lo strumento offre diverse opzioni: puoi sostituire l'intera build del gioco o modificare file specifici e puoi gestire il riavvio dei processi del server di gioco dopo gli aggiornamenti. Puoi anche usarlo per aggiornare tutti i computer di una flotta o scegliere come target singoli computer.

Visita il repository Amazon GameLift Servers Toolkit GitHub per scaricare lo [strumento di aggiornamento rapido](#) GitHub e scoprire di più su come usarlo.

Script di esempio per la distribuzione in produzione

Questo script illustra come automatizzare il processo di aggiornamento delle build di server di gioco distribuite su flotte EC2 gestite in produzione. Per utilizzare questo script, la soluzione di Amazon GameLift Servers hosting deve utilizzare alias per astrarre la flotta. Lo script di esempio gestisce i seguenti passaggi di base: caricare una build aggiornata, creare una nuova build e distribuirla in una nuova flotta, reindirizzare il traffico dei giocatori da una flotta esistente a quella nuova ed eliminare la vecchia flotta. Personalizza lo script di esempio per soddisfare i tuoi requisiti di implementazione specifici.

Visita il repository Amazon GameLift Servers Toolkit GitHub per scaricare [lo script di esempio per la distribuzione in produzione](#) GitHub e scopri di più su come usarlo.

Flotte nella console Amazon GameLift Servers

La risorsa del parco macchine rappresenta un insieme di macchine di elaborazione distribuite e gestite dai server di gioco. A seconda della soluzione di hosting, potresti avere flotte EC2 gestite, flotte di container gestite, flotte Anywhere o una combinazione.

Visualizza le informazioni sulle flotte nella Amazon GameLift Servers console o utilizzando l'SDK o l'SDK for. AWS Amazon GameLift Servers

Console

Puoi visualizzare le informazioni su tutte le flotte create per ospitare i tuoi giochi nel tuo account Amazon GameLift Servers. AWS Nella barra di navigazione a sinistra della console, trova l'opzione di hosting che desideri (Anywhere, Managed EC2, Managed containers) e scegli Fleets.

Per ogni opzione di hosting, la pagina Fleets elenca le flotte che si trovano nella regione attualmente selezionata. AWS Dalla pagina Flotte, puoi creare una nuova flotta o visualizzare ulteriori dettagli su una flotta. La [pagina dei dettagli](#) di una flotta contiene informazioni sull'utilizzo, metriche, dati sulle sessioni di gioco e dati sulle sessioni dei giocatori. Puoi anche modificare un record della flotta o eliminare una flotta.

Per impostazione predefinita, una pagina Flotte mostra le seguenti informazioni di riepilogo. Puoi modificare il contenuto della tabella secondo necessità utilizzando lo strumento Preferenze (vedi



icona

nell'angolo in alto a destra della tabella). Le preferenze personalizzate vengono salvate nell'utente AWS del tuo account e vengono applicate automaticamente ogni volta che visualizzi questa pagina.

- ID: un identificatore assegnato alla flotta. Questo ID è unico all'interno del Regione AWS luogo in cui viene creata la flotta.
- Nome: un nome descrittivo dato alla flotta.
- Stato: lo stato della flotta: Nuova, In download, In costruzione e Attiva.
- Ora di creazione: data e ora di creazione della flotta.

 Note

Un parco mostra un'icona di avviso per i parchi creati da più di 90 giorni. Come best practice, consigliamo di sostituire le flotte ogni 30 giorni per mantenere un ambiente sicuro e up-to-date di runtime per i server di gioco ospitati. Per le linee guida, consulta [Best practice relative alla sicurezza di Amazon GameLift Servers](#).

- Tipo di parco giochi: la disponibilità delle istanze utilizzate per ospitare i giochi, che può influire potenzialmente sui costi di hosting. Una flotta gestita può utilizzare istanze On-Demand (sempre disponibili) o Spot (la disponibilità varia).
- Istanze attive: il numero di istanze EC2 utilizzate per il parco istanze.
- Istanze desiderate: il numero di istanze EC2 da mantenere attive.
- Sessioni di gioco: il numero di sessioni di gioco attive in esecuzione nella flotta. I dati vengono ritardati di cinque minuti.
- Sessioni dei giocatori: il numero di sessioni di gioco attive nella flotta. I dati vengono ritardati di cinque minuti.

AWS SDK

Utilizza i seguenti comandi AWS CLI per recuperare informazioni su questa risorsa:

- Flotte EC2 gestite
 - [ListFleets](#)
 - [ListCompute](#)
 - [DescribeFleetAttributes](#)
 - [DescribeCompute](#)
 - [DescribeFleetCapacity](#)
 - [DescribeFleetUtilization](#)
- Flotte di container gestite
 - [ListContainerFleets](#)
 - [DescribeContainerFleet](#)
 - [ListContainerGroupDefinitions](#)
 - [DescribeContainerGroupDefinition](#)

- [ListFleets](#)
- [ListCompute](#)
- [DescribeFleetAttributes](#)
- [DescribeCompute](#)

Aggiornamento di una configurazione del Amazon GameLift Servers parco veicoli

Usa la Amazon GameLift Servers console o la AWS CLI per aggiornare le impostazioni del parco veicoli, modificare le postazioni remote o eliminare un parco veicoli. Per le flotte gestite, non puoi modificare la build o il tipo di istanza del server di gioco di una flotta. Invece, devi sostituire la flotta.

Fast Build Update Tool (solo per lo sviluppo)

Con EC2 le flotte gestite, per implementare un aggiornamento della build di un server di gioco, devi caricare ogni nuova build Amazon GameLift Servers e creare una nuova flotta per essa.

Il Fast Build Update Tool ti consente di aggirare questi passaggi durante lo sviluppo, risparmiando tempo e velocizzando l'iterazione dello sviluppo. Con questo strumento, puoi aggiornare rapidamente i file di build del gioco su tutti i computer di una flotta esistente. Lo strumento offre diverse opzioni: puoi sostituire un'intera build di gioco o modificare 6 file specifici e puoi gestire il riavvio dei processi del server di gioco dopo gli aggiornamenti. Puoi anche usarlo per aggiornare i singoli computer di una flotta.

Per scaricare il Fast Build Update Tool e saperne di più su come usarlo, visita il repository Amazon GameLift Servers Toolkit per [The Fast Build Update Tool](#) in Github.

È possibile aggiornare gli attributi mutabili della flotta, le impostazioni delle porte e le configurazioni di runtime utilizzando la Amazon GameLift Servers console o la AWS CLI. Per modificare i limiti di scalabilità, consulta. [Capacità di dimensionamento automatico del parco istanze con Amazon GameLift Servers](#)

Console

1. Nella [Amazon GameLift Serversconsole](#), nel riquadro di navigazione, scegli Flotte.
2. Scegli la flotta che desideri aggiornare. Una flotta deve essere in ACTIVE stato prima di poterla modificare.

3. Nella pagina dei dettagli della flotta, in una delle seguenti sezioni, scegli Modifica.
 - Impostazioni della flotta
 - Modifica gli attributi del parco istanze, ad esempio Name (Nome) e Description (Descrizione).
 - Aggiungi o rimuovi i gruppi di metriche, che Amazon CloudWatch utilizza per tenere traccia delle Amazon GameLift Servers metriche aggregate per più flotte.
 - Aggiorna le impostazioni dei limiti di creazione delle risorse.
 - Attiva o disattiva la protezione delle sessioni di gioco.
 - Configurazione di runtime: è possibile modificare una qualsiasi delle seguenti impostazioni delle configurazioni di runtime e aggiungere o rimuovere configurazioni di runtime.
 - Cambia il percorso di avvio del tuo server di gioco.
 - Aggiungi, rimuovi o modifica i parametri di avvio opzionali.
 - Modifica il numero di processi simultanei eseguiti dai server di gioco.
 - Attivazione della sessione di gioco: modifica il modo in cui desideri che i processi del server vengano eseguiti e ospitino le sessioni di gioco aggiornando il numero massimo di attivazioni simultanee delle sessioni di gioco e il nuovo timeout di attivazione.
 - EC2 impostazioni delle porte: aggiorna gli indirizzi IP e gli intervalli di porte che consentono l'accesso in entrata alla flotta.
4. Scegli Conferma per salvare le modifiche.

AWS CLI

Utilizza i seguenti comandi AWS CLI per aggiornare una flotta:

- [update-fleet-attributes](#)
- [update-fleet-port-settings](#)
- [update-runtime-configuration](#)

Aggiorna le posizioni della flotta

Puoi aggiungere o rimuovere le postazioni remote di una flotta utilizzando la Amazon GameLift Servers console o la AWS CLI. Non puoi modificare la regione di origine di una flotta.

Console

1. Nella [Amazon GameLift Serversconsole](#), nel pannello di navigazione, scegli Flotte.
2. Scegli la flotta che desideri aggiornare. Una flotta deve essere in ACTIVE stato prima di poterla modificare.
3. Nella pagina dei dettagli della flotta, scegli la scheda Sedi per visualizzare le sedi della flotta.
4. Per aggiungere nuove postazioni remote, scegli Aggiungi e seleziona le sedi in cui desideri distribuire le istanze. Questo elenco non include le istanze in cui il tipo di istanza del parco istanze non è disponibile.
5. Con le nuove sedi selezionate, scegli Aggiungi. Amazon GameLift Serversaggiunge le nuove sedi all'elenco, con lo stato impostato suNEW. Amazon GameLift Serversinizia quindi a fornire un'istanza in ogni posizione aggiunta e a prepararla per ospitare sessioni di gioco.
6. Per rimuovere le postazioni remote esistenti dal parco macchine, utilizza le caselle di controllo per selezionare una o più sedi elencate.
7. Con una o più flotte selezionate, scegli Rimuovi. Le sedi rimosse rimangono nell'elenco, con lo stato impostato DELETING su. Amazon GameLift Serversinizia quindi il processo di cessazione dell'attività nella località rimossa. Se ci sono istanze attive che ospitano sessioni di gioco, Amazon GameLift Servers utilizza la procedura di chiusura del server di gioco per terminare correttamente le sessioni di gioco, terminare i server di gioco e chiudere le istanze.

AWS CLI

Utilizza i seguenti comandi AWS CLI per aggiornare le ubicazioni del parco veicoli:

- [create-fleet-locations](#)
- [delete-fleet-locations](#)

Eliminare un parco veicoli

Puoi eliminare una flotta quando non ne hai più bisogno. L'eliminazione di una flotta rimuove definitivamente tutti i dati associati alle sessioni di gioco e alle sessioni dei giocatori e i dati metrici raccolti. In alternativa, puoi mantenere la flotta, disabilitare la scalabilità automatica e ridimensionare manualmente la flotta a 0 istanze.

 Note

Se la flotta dispone di una connessione peering VPC, richiedi prima l'autorizzazione chiamando [CreateVpcPeeringAuthorization](#) Amazon GameLift Servers. Elimina la connessione peering VPC durante l'eliminazione del parco veicoli.

Puoi utilizzare la Amazon GameLift Servers console o lo strumento AWS CLI per eliminare una flotta.

Console

1. Nella [Amazon GameLift Servers console](#), nel riquadro di navigazione, scegli Flotte.
2. Scegli la flotta che desideri eliminare. Puoi eliminare solo le flotte presenti nel ACTIVE nostro ERROR stato.
3. Scegli Elimina.
4. Nella finestra di dialogo Elimina flotta, conferma l'eliminazione **delete** inserendo.
5. Scegli Elimina.

AWS CLI

Utilizza il seguente comando AWS CLI per eliminare una flotta:

- [delete-fleet](#)

Dettagli della flotta nella Amazon GameLift Servers console

Accedi a una pagina dei dettagli della flotta dalla dashboard o dalla pagina Flotte scegliendo il nome della flotta.

Nella pagina dei dettagli della flotta puoi intraprendere le seguenti azioni:

- Aggiorna gli attributi, le impostazioni delle porte e la configurazione di runtime di una flotta.
- Aggiungi o rimuovi le sedi del parco veicoli.
- Modificare le impostazioni della capacità del parco istanze.
- Imposta o modifica il ridimensionamento automatico del tracciamento del bersaglio.
- Eliminare un parco istanze.

Informazioni

Impostazioni della flotta

- **Fleet ID:** un identificatore assegnato alla flotta. Questo ID è unico all'interno del Regione AWS luogo in cui viene creata la flotta.
- **Nome della flotta:** un nome descrittivo assegnato alla flotta.
- **ARN:** un identificatore univoco assegnato a questa flotta. L'ARN di una flotta la identifica come Amazon GameLift Servers risorsa e specifica la regione e l'account. AWS
- **Descrizione:** una breve descrizione identificabile della flotta.
- **Stato:** stato attuale della flotta, che può essere Nuovo, In download, In costruzione e Attivo.
- **Ora di creazione:** data e ora di creazione della flotta.

Note

Un parco mostra un'icona di avviso per i parchi creati da più di 90 giorni. Come best practice, consigliamo di sostituire le flotte ogni 30 giorni per mantenere un ambiente sicuro e up-to-date di runtime per i server di gioco ospitati. Per le linee guida, consulta [Best practice relative alla sicurezza di Amazon GameLift Servers](#).

- **Tipo di elaborazione:** il tipo di elaborazione utilizzato per ospitare i tuoi giochi. Una flotta può essere una flotta di container gestita EC2, gestita o Anywhere.
- **Sistema operativo:** Amazon Machine Image (AMI) che viene distribuito su ogni istanza del parco istanze. La versione AMI è l'ultima versione disponibile al momento della creazione della flotta.
- **Tipo di certificato:** lo stato della capacità della flotta di utilizzare un certificato TLS per autenticare un server di gioco e crittografare tutte le comunicazioni. client/server
- **Tipo di parco giochi:** la disponibilità delle istanze utilizzate per ospitare i giochi, che può influire potenzialmente sui costi di hosting. Un parco istanze può utilizzare istanze On-Demand (sempre disponibili) o Spot (la disponibilità varia).
- **EC2 tipo di istanza:** [tipo di EC2 istanza](#) Amazon selezionato per la flotta al momento della creazione.
- **ARN del ruolo dell'istanza:** un ruolo AWS IAM che gestisce l'accesso alle altre AWS risorse, se ne è stata fornita una durante la creazione della flotta.
- **Provider di credenziali per il ruolo dell'istanza:** un ruolo AWS IAM che gestisce l'accesso alle altre AWS risorse, se ne è stato fornito uno durante la creazione della flotta.

- Nome del gruppo di metriche: il gruppo utilizzato per aggregare le metriche per più flotte.
- Politica di protezione Game Scaling: indica se la protezione delle sessioni di gioco della flotta è abilitata, il che impedisce che le sessioni di gioco attive terminino prematuramente durante un evento di ridimensionamento.
- Numero massimo di sessioni di gioco per giocatore: il numero massimo di sessioni che un giocatore può creare durante il periodo di validità della Politica.
- Periodo di validità: il periodo di tempo utilizzato per limitare il numero di sessioni create da un giocatore.

Dettagli della costruzione

La sezione Dettagli di costruzione mostra la build ospitata nella flotta. Seleziona il nome della build per visualizzare la pagina di dettaglio completa della build.

Configurazione di runtime

La sezione di configurazione del runtime mostra i processi del server da avviare su ciascuna istanza. Include il percorso dei file eseguibili del server di gioco e i parametri opzionali di avvio.

Attivazione della sessione di gioco

La sezione Attivazione della sessione di gioco mostra il numero di processi del server che vengono avviati contemporaneamente e per quanto tempo attendere l'attivazione del processo prima di terminarlo.

EC2 impostazioni delle porte

La sezione Porte mostra le autorizzazioni di connessione della flotta, inclusi gli indirizzi IP e gli intervalli di impostazione delle porte.

Metriche

Nella scheda Parametri è visualizzata una rappresentazione grafica dei parametri del parco istanze nel tempo. Per ulteriori informazioni sull'utilizzo delle metriche in Amazon GameLift Servers, consulta [Monitora Amazon GameLift Servers con Amazon CloudWatch](#)

Eventi

La scheda Events (Eventi) fornisce un log di tutti gli eventi che si sono verificati nel parco istanze, incluso il codice dell'evento, il messaggio e il timestamp. Vedi le descrizioni degli [eventi](#) nell'Amazon GameLift Servers API Reference.

Dimensionamento

La scheda Scaling contiene informazioni sulla capacità della flotta, tra cui lo stato attuale e le variazioni di capacità nel tempo. Fornisce inoltre strumenti per aggiornare i limiti di capacità e gestire la scalabilità automatica.

Capacità di scalabilità

Visualizza le impostazioni correnti della capacità del parco veicoli per ogni sede del parco veicoli. Per ulteriori informazioni sulla modifica dei limiti e della capacità, consulta [Scalabilità della capacità di hosting di giochi con Amazon GameLift Servers](#).

- **AWS Ubicazione:** nome di una sede in cui vengono distribuite le istanze della flotta.
- **Stato:** lo stato di hosting della posizione del parco. Lo stato della posizione deve essere ACTIVE per poter ospitare le partite.
- **Dimensione minima:** il numero minimo di istanze che devono essere distribuite nella posizione.
- **Istanze desiderate:** il numero target di istanze attive per mantenere la posizione. Quando le istanze attive e le istanze desiderate non coincidono, viene avviato un evento di scalabilità per avviare o chiudere le istanze secondo necessità fino a quando le istanze attive non raggiungono le istanze desiderate.
- **Dimensione massima:** il maggior numero di istanze che possono essere distribuite nella posizione.
- **Disponibile:** il limite di servizio per le istanze meno il numero di istanze in uso. Questo valore indica il numero massimo di istanze che è possibile aggiungere alla posizione.

Policy di scalabilità automatica

Questa sezione contiene informazioni sulle politiche di scalabilità automatica applicate al parco veicoli. È possibile configurare o aggiornare una politica basata sugli obiettivi. Le politiche basate su regole della flotta, che devono essere definite utilizzando l' AWS SDK o la CLI, vengono visualizzate qui. Per ulteriori informazioni sul dimensionamento, consulta [Capacità di dimensionamento automatico del parco istanze con Amazon GameLift Servers](#).

Cronologia della scalabilità

Visualizza i grafici delle variazioni di capacità nel tempo.

Posizioni

La scheda Sedi elenca tutte le sedi in cui vengono distribuite le istanze del parco istanze. Le sedi includono la regione di origine della flotta e tutte le postazioni remote che sono state aggiunte. È possibile aggiungere o rimuovere le posizioni direttamente in questa scheda.

- **Ubicazione:** nome di una sede in cui vengono distribuite le istanze del parco istanze.
- **Stato:** lo stato di hosting della posizione del parco. Lo stato della posizione tiene traccia del processo di attivazione delle prime istanze nella posizione. Lo stato della posizione deve essere **ACTIVE** per poter ospitare le partite.
- **Istanze attive:** il numero di istanze con processi server in esecuzione nella posizione del parco.
- **Server attivi:** il numero di processi del server di gioco in grado di ospitare sessioni di gioco nella posizione del parco.
- **Sessioni di gioco:** il numero di sessioni di gioco attive sulle istanze presenti nella sede del parco istanze.
- **Sessioni dei giocatori:** il numero di sessioni dei giocatori, che rappresentano singoli giocatori, che partecipano a sessioni di gioco attive nella sede della flotta.

Sessioni di gioco

La scheda Game sessions (Sessioni di gioco) mostra le sessioni di gioco passate e presenti ospitate nel parco istanze, oltre ad alcune informazioni dettagliate. Scegli un ID di sessione di gioco per accedere a informazioni aggiuntive sulla sessione di gioco, incluse le sessioni dei giocatori. Per ulteriori informazioni sulle sessioni dei giocatori, consulta [Sessioni di gioco e di gioco nella Amazon GameLift Servers console](#).

Sessioni di gioco e di gioco nella Amazon GameLift Servers console

Puoi utilizzare la Amazon GameLift Servers console per lavorare con sessioni di gioco e sessioni di gioco. Per ulteriori informazioni sulle sessioni di gioco e sulle sessioni giocatore, consulta [Amazon GameLift Serverse l'esperienza del giocatore](#). La Amazon GameLift Servers console fornisce informazioni e strumenti per aiutarti a risolvere i problemi relativi alle sessioni di gioco.

Cosa puoi fare:

- Esplora le attività delle sessioni di gioco e delle sessioni dei giocatori ospitate su una flotta specifica.
- Cerca le attività delle sessioni di gioco per un giocatore specifico su più flotte.

- Chiudi una sessione di gioco specifica.

Visualizza i dettagli della sessione di gioco

I dati della sessione di gioco e della sessione dei giocatori sono organizzati dalla flotta che ospita la sessione di gioco.

Per accedere alle informazioni sulla sessione di gioco e sulla sessione del giocatore

1. Nella [Amazon GameLift Servers console](#), apri il riquadro di navigazione a sinistra. Seleziona un tipo di soluzione di hosting e apri la pagina Fleets. Esempio:
 - Hosting, ovunque, di flotte
 - Hosting, EC2 gestito, flotte
 - Hosting, contenitori gestiti, flotte
2. Ogni pagina Flotte mostra l'elenco delle flotte attualmente selezionate. Regione AWS Scegli la flotta di cui desideri visualizzare i dati della sessione di gioco.
3. Nella pagina dei dettagli della flotta, apri la scheda Sessioni di gioco. Questa scheda elenca tutte le sessioni di gioco ospitate sulla flotta, insieme a informazioni di riepilogo. Puoi modificare il contenuto della tabella in base alle tue esigenze utilizzando lo strumento Preferenze (vedi  icona nell'angolo in alto a destra della tabella). Le preferenze personalizzate vengono salvate nell'utente AWS del tuo account e vengono applicate automaticamente ogni volta che visualizzi questa pagina.
4. Scegli una sessione di gioco dall'elenco per visualizzare informazioni aggiuntive.
5. Se la sessione di gioco include i dati delle sessioni dei giocatori, scegli Visualizza le sessioni dei giocatori per aprire lo strumento di ricerca delle sessioni dei giocatori con l'ID della sessione di gioco inserito automaticamente.

I dettagli delle sessioni di gioco includono le seguenti informazioni:

- Stato: stato della sessione di gioco.
 - Attivazione: l'istanza sta avviando una sessione di gioco.
 - Attiva: una sessione di gioco è in corso ed è disponibile per ricevere giocatori, a seconda della politica di [creazione dei giocatori](#) della sessione.

- Terminata: la sessione di gioco è terminata.
- ARN: il nome Amazon Resource della sessione di gioco.
- Nome: nome generato per la sessione di gioco.
- Luogo: il luogo in cui è Amazon GameLift Servers stata ospitata la sessione di gioco.
- Ora di creazione: data e ora di Amazon GameLift Servers creazione della sessione di streaming.
- Ora di fine: data e ora in cui è terminata la sessione di gioco.
- Nome DNS: il nome host della sessione di gioco.
- Indirizzo IP: indirizzo IP specificato per la sessione di gioco.
- Porta: numero di porta utilizzato per connettersi alla sessione di gioco.
- Creator ID: un identificativo univoco del giocatore che ha avviato la sessione di gioco.
- Politica di creazione della sessione dei giocatori: indica se la sessione di gioco accetta nuovi giocatori.
- Politica di protezione della scalabilità del gioco: il tipo di protezione della sessione di gioco da impostare su tutte le nuove istanze che entrano a far Amazon GameLift Servers parte della flotta.

Dati di gioco

Dati sulle proprietà del gioco, formattati come stringa, da inviare alla sessione di gioco all'inizio.

Proprietà del gioco

Dati sulle proprietà del gioco, formattati come key/value coppie, da inviare alla sessione di gioco all'inizio.

Dati di matchmaking

Se la sessione di gioco è stata creata con FlexMatch, i dati di matchmaking descrivono le informazioni sulla configurazione del matchmaking e sul set di regole. Ciò include gli attributi dei giocatori e gli incarichi di squadra di ogni partita. I dati sono in formato JSON. Per ulteriori informazioni sul FlexMatch matchmaking, vedi [Build a matchmaker](#).

Look up player session data

Se la tua soluzione di hosting di giochi utilizza sessioni di gioco e offre giocatori unici IDs, puoi esplorare le attività specifiche del giocatore per le sessioni di gioco passate o presenti su più flotte. Apri lo strumento di ricerca delle sessioni Player utilizzando uno dei seguenti metodi:

- Nella Amazon GameLift Servers console, apri il riquadro di navigazione a sinistra e scegli Player session lookup e seleziona il tipo di filtro da utilizzare.
- Quando visualizzi i dettagli della sessione di gioco di una flotta, scegli Visualizza le sessioni dei giocatori. Lo strumento di ricerca si apre con la sessione di gioco con il filtro ID della sessione di gioco preselezionato e il valore della sessione di gioco inserito.

Quando utilizzi lo strumento di ricerca, puoi fornire le seguenti informazioni:

- Un ID di sessione del giocatore per ottenere informazioni su una sessione di gioco specifica.
- Un ID di sessione di gioco per ottenere informazioni su tutte le sessioni dei giocatori per la sessione di gioco richiesta. I risultati rappresentano tutti i giocatori che hanno prenotato uno slot o si sono collegati alla sessione di gioco. È possibile facoltativamente filtrare i risultati in base allo stato della sessione del giocatore.
- Un ID giocatore per ottenere informazioni su tutte le sessioni di gioco per il giocatore richiesto. I risultati rappresentano tutte le sessioni di gioco a cui ha partecipato il giocatore.

Note

Lo strumento di ricerca cerca tutte le attività delle sessioni dei giocatori tra quelle attualmente selezionate. Regione AWS Se hai più flotte nella Regione, i risultati includono l'attività delle sessioni dei giocatori in tutte le flotte. Per le flotte con più sedi, i risultati includono anche l'attività delle sessioni dei giocatori nelle località remote delle flotte.

Per ciascuna sessione di gioco vengono raccolti i seguenti dati delle sessioni giocatore:

- ID della sessione del giocatore: l'identificatore assegnato alla sessione del giocatore.
- ID giocatore: un identificatore univoco per il giocatore. Scegli questo ID per ottenere ulteriori informazioni sul giocatore.
- ID della sessione di gioco: l'identificatore assegnato alla sessione di gioco.
- Fleet ID: l'identificativo assegnato alla flotta che ha ospitato la sessione di gioco.
- Stato: lo stato della sessione del giocatore. Gli stati possibili sono i seguenti:
 - Riservata: la sessione del giocatore è stata riservata, ma i giocatori non sono connessi.
 - Attiva: la sessione del giocatore è connessa al server di gioco.

- **Completata:** la sessione del giocatore è terminata; il giocatore non è più connesso.
- **Timeout:** il giocatore non è riuscito a connettersi.
- **Ora di creazione:** l'ora in cui il giocatore si è connesso alla sessione di gioco.
- **Ora di fine:** l'ora in cui il giocatore si è disconnesso dalla sessione di gioco.
- **Dati di connessione:** indirizzo IP, nome DNS e porta utilizzati dal giocatore per connettersi alla sessione di gioco.
- **Dati del giocatore:** informazioni sul giocatore fornite durante la creazione della sessione di gioco.

Shut down a game session

Usa la Amazon GameLift Servers console per chiudere una sessione di gioco specifica. Questa funzione offre un metodo semplice e veloce per localizzare una sessione di gioco e inviare un segnale per terminarla. Un altro metodo di terminazione richiede di trovare l'istanza Fleet su cui è in corso la sessione di gioco, accedere in remoto all'istanza e chiudere manualmente la sessione di gioco.

Puoi interrompere una sessione di gioco per qualsiasi motivo. Il motivo più comune è risolvere una sessione di gioco che non si chiude naturalmente. Di conseguenza, la risorsa di hosting per la sessione di gioco non può essere liberata per ospitare una nuova sessione di gioco e la capacità di hosting della flotta viene ridotta.

Note

Questa funzionalità si basa su determinate impostazioni di configurazione per la tua soluzione di hosting. Presenta le seguenti limitazioni:

- La sessione di gioco deve essere ospitata su una flotta che esegue un server di gioco costruito con server SDK per Amazon GameLift Servers v5 o versioni successive. Se i tuoi server di gioco utilizzano una versione precedente, devi utilizzare l'accesso remoto per eliminare la sessione di gioco.
- Se la sessione di gioco è ospitata su una flotta Anywhere, la flotta deve utilizzare l'Amazon GameLift Servers agente per gestire i processi del server di gioco.

Per terminare una sessione di gioco

1. Nella [Amazon GameLift Servers console](#), apri il riquadro di navigazione a sinistra. Seleziona un tipo di soluzione di hosting e apri la pagina Fleets. Esempio:
 - Hosting, ovunque, di flotte
 - Hosting, EC2 gestito, flotte
 - Hosting, contenitori gestiti, flotte
2. Ogni pagina Flotte mostra l'elenco delle flotte attualmente selezionate. Regione AWS Scegli la flotta che ospita la sessione di gioco che desideri interrompere.
3. Nella pagina dei dettagli della flotta, apri la scheda Sessioni di gioco. Nell'elenco delle sessioni di gioco, seleziona quella che desideri terminare e scegli il pulsante Termina.
4. Nella sessione di gioco Termina? nella finestra, verifica di chiudere la sessione di gioco corretta e scegli un metodo di terminazione.
 - Chiusura normale della sessione di gioco: questa opzione invia un segnale di chiusura al processo del server che ospita la sessione di gioco. Se la build del server di gioco è stata integrata correttamente per Amazon GameLift Servers, il processo server avvia la sequenza di chiusura della sessione di gioco, notifica la fine della sessione e si Amazon GameLift Servers interrompe. A seconda del design del gioco, la sequenza di spegnimento potrebbe includere passaggi per completare correttamente la sessione di gioco, come il salvataggio dei dati e la notifica ai giocatori attivi. Questo metodo potrebbe richiedere un piccolo ritardo per completare la sequenza di chiusura della sessione di gioco.
 - Chiusura immediata della sessione di gioco: questa opzione invia un segnale a un gestore dei processi per chiudere il processo del server che ospita la sessione di gioco. Questa opzione ignora la normale chiusura della sessione di gioco. È in grado di terminare la sessione di gioco anche quando il processo del server non è in grado di rispondere.
5. Conferma la cessazione della sessione di gioco. Puoi tenere traccia dell'avanzamento della chiusura nella pagina della console delle sessioni di gioco. Lo stato della sessione di gioco cambierà in «Interruzione» e poi in «Interrotta» al termine della sessione di gioco.

Argomenti correlati

- Puoi anche chiudere le sessioni di gioco utilizzando l'SDK e il. AWS CLI Per ulteriori dettagli ed esempi, consulta l'argomento Amazon GameLift Servers API Reference.

[TerminateGameSession](#)

- Per ulteriori informazioni sull'integrazione dei server di gioco e su come un processo server risponde ai segnali del Amazon GameLift Servers servizio, consulta [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#).

Aggiorna una flotta di container Amazon GameLift Servers gestita

È possibile aggiornare la maggior parte delle proprietà di una flotta di container gestita, incluse le definizioni dei gruppi di container. A seconda delle impostazioni da aggiornare, un aggiornamento del parco veicoli potrebbe avviare una nuova implementazione del parco veicoli. In una distribuzione con un parco istanze, tutte le istanze del parco istanze vengono rimosse e sostituite con istanze con la nuova configurazione. Le impostazioni che richiedono una distribuzione includono:

- Definizioni dei gruppi di contenitori, inclusi gli aggiornamenti alle immagini dei contenitori
- Intervalli di porte di connessione e autorizzazioni in ingresso
- Configurazione del registro

Puoi tenere traccia dello stato delle implementazioni della flotta nella [Amazon GameLift Serversconsole](#) o AWS Command Line Interface (AWS CLI) per creare una flotta di container.

Note

Non puoi aggiornare l'ambiente di runtime di una flotta. Al momento della creazione della flotta, l'Amazon Machine Image (AMI) della flotta è impostata sull'ultima versione disponibile dell'AMI Linux. Tutte le immagini dei container distribuite in questa flotta devono essere compatibili con questa versione. Per modificare l'AMI della flotta o passare a una versione più recente, devi creare una nuova flotta. Come best practice, consigliamo di sostituire le flotte ogni 30 giorni per mantenere un ambiente sicuro e di up-to-date runtime per i server di gioco ospitati. Per ulteriori informazioni, consulta [Best practice relative alla sicurezza di Amazon GameLift Servers](#).

Console

Nella [Amazon GameLift Serversconsole](#), seleziona il Regione AWS punto in cui desideri creare la flotta. Le definizioni dei gruppi di container devono trovarsi nella stessa regione in cui desideri creare la flotta.

Apri la barra di navigazione a sinistra della console e scegli Contenitori gestiti: flotte. Nella pagina Flotte di container gestite, seleziona una flotta dall'elenco e scegli Modifica.

1. Aggiorna le impostazioni della flotta di container secondo necessità. Al termine, scegliere Create (Crea).
2. Se gli aggiornamenti richiedono l'implementazione di una flotta, ti viene chiesto di specificare le opzioni di distribuzione come segue:
 - Protezione delle sessioni di gioco. Puoi scegliere di proteggere le istanze della flotta con sessioni di gioco attive (distribuzione sicura). Con questa impostazione, le istanze della flotta vengono sostituite solo dopo la fine delle sessioni di gioco. In alternativa, puoi scegliere di sostituire le istanze della flotta indipendentemente dall'attività della sessione di gioco (distribuzione non sicura). Le implementazioni non sicure sono utili durante le fasi di sviluppo e test per ridurre i tempi di implementazione.
 - Percentuale minima di salute. Puoi gestire la velocità con cui le istanze del parco istanze vengono sostituite. Utilizza questa impostazione per mantenere un numero minimo di attività integre durante la distribuzione. Un valore basso dà priorità alla velocità di distribuzione, mentre un valore alto assicura che la disponibilità del server di gioco rimanga elevata per tutta la durata della distribuzione.
 - Strategia di fallimento dell'implementazione. Decidi quali azioni intraprendere se una distribuzione fallisce. Un errore di distribuzione significa che alcuni contenitori aggiornati non hanno superato i controlli di stato e sono considerati compromessi. È possibile impostare le distribuzioni per ripristinare automaticamente tutte le istanze del parco istanze allo stato di distribuzione precedente. In alternativa, puoi scegliere di mantenere alcune delle istanze del parco istanze danneggiate da utilizzare nel debug.

Se la richiesta ha esito positivo, la console visualizza la scheda Distribuzioni per la flotta di container gestita. Utilizza questa scheda per tenere traccia dello stato di ogni implementazione. Se avvii una nuova distribuzione per la flotta, questa azione annulla automaticamente qualsiasi distribuzione attualmente in corso per la flotta.

AWS CLI

Per creare una flotta di container con AWS CLI, apri una finestra della riga di comando e usa il `update-container-fleet` comando. Per ulteriori informazioni su questo comando, vedere [update-container-fleet](#) nella Guida ai AWS CLI comandi.

L'esempio seguente aggiorna una flotta di container esistente con le seguenti caratteristiche:

- Aggiorna la definizione del gruppo di contenitori del server di gioco per utilizzare la versione 2.
- Specifica le opzioni di distribuzione sicure.

```
{
  "DeploymentConfiguration": {
    "ImpairmentStrategy": "ROLLBACK",
    "MinimumHealthyPercentage": 75,
    "ProtectionStrategy": "WITH_PROTECTION"
  },
  "FleetId": "containerfleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
  "GameServerContainerGroupDefinitionName": "arn:aws:gamelift:us-west-2:111122223333:containergroupdefinition/MyAdventureGameContainerGroup:2"
}
```

Aggiornare una definizione di gruppo di container per una flotta di Amazon GameLift Servers container

È possibile aggiornare la maggior parte delle proprietà di una definizione di gruppo di contenitori, incluse le definizioni dei singoli contenitori. Le definizioni dei gruppi di contenitori hanno un numero di versione. Quando aggiorni una definizione di gruppo di contenitori, Amazon GameLift Servers salva l'aggiornamento e incrementa il numero di versione della definizione. Quando si configura una flotta di container, è possibile specificare quale versione della definizione di gruppo di contenitori distribuire.

Dopo aver aggiornato la definizione di un gruppo di container, puoi distribuire la nuova versione su una flotta di container nuova o esistente.

Aggiorna la definizione del gruppo di contenitori di un server di gioco

Questo argomento descrive come aggiornare la definizione del gruppo di contenitori del server di gioco utilizzando gli strumenti della Amazon GameLift Servers console o della AWS CLI. Per informazioni più dettagliate sulle funzionalità opzionali, consulta [Personalizza una flotta di Amazon GameLift Servers container](#).

Per aggiornare la definizione di un gruppo di contenitori:

Console

Nella [Amazon GameLift Servers console](#), seleziona il Regione AWS punto in cui desideri creare il gruppo di contenitori.

Apri la barra di navigazione a sinistra della console e scegli Contenitori gestiti: definizioni di gruppo. Nella pagina di definizione dei gruppi di contenitori, scegli una definizione e una versione del gruppo di contenitori da aggiornare.

Dopo aver salvato gli aggiornamenti, puoi utilizzare la nuova versione per creare nuove flotte di container oppure puoi distribuire gli aggiornamenti a una flotta di container esistente.

Fase 1: Definire i dettagli della definizione del gruppo di contenitori

- È possibile aggiornare le impostazioni dei limiti di memoria totale e vCPU.

Fase 2: Aggiungere le definizioni dei contenitori

È possibile apportare i seguenti aggiornamenti delle definizioni dei contenitori:

- Aggiorna le definizioni dei contenitori esistenti.
 - Aggiungi nuove definizioni dei contenitori di supporto.
 - Rimuovi le definizioni dei contenitori di supporto.
1. È possibile aggiornare l'URI dell'immagine ECR. Assicurati di aggiornare l'impostazione della versione di Server SDK in modo che corrisponda alla nuova immagine.
 2. È possibile aggiornare l'intervallo di porte interne del contenitore in base alle esigenze. Le modifiche apportate a queste impostazioni potrebbero influire sulle impostazioni delle porte di connessione di una flotta di container quando queste modifiche vengono implementate in una flotta. Per ulteriori dettagli, consultare [Configurare le connessioni di rete](#).

Fase 3: Configurare le dipendenze

- È possibile modificare le dipendenze in base alle esigenze. Per ulteriori informazioni, consulta [Imposta le dipendenze del container](#).

Fase 3: Revisione e creazione

- Controlla gli aggiornamenti delle definizioni dei gruppi di contenitori. Usa Modifica per apportare ulteriori modifiche in qualsiasi sezione. Al termine, scegli Crea per generare una nuova versione della definizione del gruppo di contenitori.

Se la richiesta ha esito positivo, la console visualizza la pagina di dettaglio per la nuova risorsa di definizione del gruppo di contenitori. Inizialmente lo stato è `COPYING`: Amazon GameLift Servers inizia a scattare istantanee di tutte le immagini del contenitore per il gruppo. Al termine di questa fase, lo stato della definizione del gruppo di contenitori cambia in `READY`. Una definizione di gruppo di container deve essere `READY` attiva prima di poter creare una flotta di container con essa.

AWS CLI

Quando utilizzi la AWS CLI per creare o aggiornare una definizione di gruppo di contenitori, mantieni le configurazioni della definizione del contenitore in un file separato. JSON Puoi fare riferimento al file nel tuo comando CLI. Vedi alcuni esempi [Crea un file di definizione del contenitore JSON](#) di schema.

Quando aggiorni una definizione, devi solo specificare i valori che desideri aggiornare. Amazon GameLift Servers conserva tutti i valori che non includi nella richiesta di aggiornamento. Se stai modificando la definizione di un contenitore. Tuttavia, quando modificate la definizione di un contenitore, fornite un set completo.

Per aggiornare una definizione di gruppo di contenitori

Per aggiornare una nuova definizione di gruppo di contenitori, utilizzate il `update-container-group-definition` comando CLI. Per ulteriori informazioni su questo comando, [update-container-group-definition](#) consultate la AWS CLI Command Reference.

Example: gruppo di contenitori per server di gioco

È possibile specificare una versione della definizione del gruppo di contenitori durante il recupero, l'aggiornamento o l'eliminazione di una definizione del gruppo di contenitori o durante la creazione o l'aggiornamento di una flotta di container. Ogni definizione di gruppo di contenitori ha una proprietà di versione. Inoltre, il valore ARN della definizione `and` specifica il numero di versione.

Questo esempio illustra una richiesta di modifica alla definizione di un gruppo di contenitori di server di gioco. Si presuppone che tu abbia creato un file JSON con le definizioni dei contenitori per questo gruppo. Questo esempio utilizza il valore ARN per il nome della definizione e specifica che l'aggiornamento è alla versione 1.

```
aws gamelift update-container-group-definition \
```

```
--name arn:aws:gamelift:us-west-2:111122223333:containergroupdefinition/
MyAdventureGameContainerGroup:1 \
--operating-system AMAZON_LINUX_2023 \
--container-group-type GAME_SERVER \
--total-memory-limit-mebibytes 4096 \
--total-vcpu-limit 1 \
--container-definitions file://SimpleServer.json
```

Clonare una definizione di gruppo di contenitori

È possibile utilizzare la Amazon GameLift Servers console per clonare una definizione di gruppo di contenitori esistente.

Per clonare un gruppo di contenitori

1. Nella [Amazon GameLift Servers console](#), vai al riquadro di navigazione a sinistra e scegli Gruppi di contenitori.
2. Nella pagina di elenco dei gruppi di contenitori, seleziona il gruppo di contenitori esistente che desideri clonare. Dopo aver selezionato un gruppo di contenitori, il pulsante Clona è attivo.
3. Seleziona Clona. Questa azione apre la procedura guidata per la creazione di gruppi di contenitori con impostazioni precompilate.
4. Immettete un nuovo nome per il gruppo di contenitori clonato. Il gruppo di contenitori nella stessa regione deve avere nomi univoci.
5. Scorri le pagine del gruppo di contenitori e delle definizioni dei contenitori, esamina e crea il nuovo gruppo di contenitori.

Eliminare una definizione di gruppo di container per una flotta di Amazon GameLift Servers container

Sono disponibili diverse opzioni per eliminare una definizione di gruppo di contenitori. Quando si elimina una definizione di gruppo di contenitori, questa azione elimina anche tutte le definizioni di contenitori nel gruppo di contenitori.

Le definizioni dei gruppi di contenitori possono avere più versioni. Le versioni dei gruppi di contenitori hanno lo stesso nome, ma hanno un numero di versione diverso. La definizione del gruppo di contenitori ARNs specifica sia il nome che la versione.

È possibile specificare una versione della definizione del gruppo di contenitori durante il recupero, l'aggiornamento o l'eliminazione di una definizione di gruppo di contenitori o durante la creazione o l'aggiornamento di una flotta di contenitori. Ogni definizione di gruppo di contenitori ha una proprietà di versione. Inoltre, il valore ARN della definizione specifica il numero di versione.

Esistono diversi modi per eliminare le definizioni dei gruppi di contenitori:

- È possibile eliminare tutte le versioni di una definizione specifica.
- È possibile eliminare una versione particolare di una definizione specifica.
- È possibile conservare un certo numero di versioni più recenti ed eliminare le versioni precedenti di una definizione specifica. Ad esempio, è possibile eliminare tutte le versioni precedenti alla versione 5.

È possibile eliminare una versione della definizione del gruppo di contenitori solo se non viene utilizzata in una flotta di container.

Per eliminare una definizione di gruppo di contenitori

Console

Nella [Amazon GameLift Serversconsole](#), seleziona il Regione AWS punto in cui desideri creare il gruppo di contenitori.

Apri la barra di navigazione a sinistra della console e scegli Contenitori gestiti: definizioni di gruppo. Nella pagina Definizioni dei gruppi di contenitori, seleziona la definizione che desideri modificare e scegli Elimina.

Ti viene richiesto di selezionare il tipo di eliminazione che desideri effettuare e di specificare impostazioni aggiuntive a seconda del tipo di eliminazione.

AWS CLI

- Per eliminare una definizione di gruppo di contenitori, utilizza il comando `delete-container-group-definition` CLI e fornisci i valori per il tipo di eliminazione che desideri eseguire. Per ulteriori informazioni su questo comando, [delete-container-group-definition](#)consultate la AWS CLI Command Reference.

Questo esempio illustra una richiesta di eliminazione di tutte le versioni della definizione del gruppo di contenitori del server di gioco precedenti alla versione 5.

Example

```
aws gamelift delete-container-group-definition \  
  --name MyAdventureGameContainerGroup \  
  --version-count-to-retain 5 \  

```

Alias nella console Amazon GameLift Servers

La risorsa alias rappresenta un puntatore a una flotta o a una destinazione terminale. Vengono utilizzati per indirizzare il traffico dei giocatori verso destinazioni di hosting specifiche. Usa un alias per astrarre una destinazione in modo da poter reindirizzare più facilmente il traffico quando aggiorni le flotte di hosting.

Visualizza le informazioni sugli alias nella Amazon GameLift Servers console o utilizzando l'SDK o l'SDK per. AWS Amazon GameLift Servers

Console

La pagina Alias mostra informazioni sugli Amazon GameLift Servers alias che indirizzano il traffico verso destinazioni di hosting specifiche. Per visualizzare gli alias, scegli Hosting, Alias nel pannello di navigazione.

Nella pagina degli alias puoi effettuare le seguenti operazioni:

- Crea un nuovo alias. Scegli Crea alias.
- Filtra e ordina la tabella degli alias. Utilizzare i controlli nella parte superiore della tabella. Puoi modificare il contenuto della tabella secondo necessità utilizzando lo strumento Preferenze (vedi  icona nell'angolo in alto a destra della tabella). Le preferenze personalizzate vengono salvate nell'utente AWS del tuo account e vengono applicate automaticamente ogni volta che visualizzi questa pagina.
- Visualizzare i dettagli degli alias. Scegli un nome alias per aprire la pagina dei dettagli dell'alias.
- Eliminare un alias. Scegli un alias, quindi scegli Elimina.

Dettagli dell'alias

La pagina dei dettagli dell'alias mostra informazioni sull'alias.

In questa pagina, è possibile:

- Modifica un alias. Scegli Modifica.
- Visualizza le flotte che hai associato all'alias.
- Eliminare un alias. Scegli Elimina.

Le informazioni dettagliate sugli alias includono:

- ID: il numero univoco utilizzato per identificare l'alias.
- Descrizione: la descrizione dell'alias.
- ARN: il nome Amazon Resource dell'alias.
- Creazione: data e ora di creazione dell'alias.
- Ultimo aggiornamento: la data e l'ora dell'ultimo aggiornamento dell'alias.
- Tipo di routing: il tipo di routing per l'alias, che può essere uno dei seguenti:
 - Semplice: indirizza il traffico dei giocatori verso un ID flotta specificato. È possibile aggiornare l'ID del parco istanze per un alias in qualsiasi momento.
 - Terminale: restituisce un messaggio al client. Ad esempio, puoi indirizzare i giocatori che utilizzano un out-of-date client verso un luogo in cui possono ottenere un upgrade.
- Tag: coppie di chiavi e valori utilizzate per identificare l'alias.

AWS SDK

Utilizza i seguenti comandi AWS CLI per recuperare informazioni su questa risorsa:

- [ListAliases](#)
- [DescribeAlias](#)

Modificare un alias

È possibile modificare un alias utilizzando Amazon GameLift Servers [console o con il comando AWS CLI update-alias](#).

Questo argomento descrive come modificare un Amazon GameLift Servers alias da utilizzare per il posizionamento delle sessioni di gioco. Puoi apportare le seguenti modifiche:

Per modificare un alias

Usa uno dei Amazon GameLift Servers console o AWS Command Line Interface (AWS CLI) per modificare un alias.

Console

Nella [Amazon GameLift Servers console](#), usa il pannello di navigazione per aprire la pagina Alias.

1. Seleziona l'alias che desideri modificare e scegli Modifica. Se non vedi l'alias che desideri modificare, controlla quello attualmente selezionato. Regione AWS
2. Nella pagina Modifica alias, puoi apportare le seguenti modifiche:
 - Cambia il nome dell'alias.
 - Modificare la descrizione dell'alias.
 - Cambia la strategia di routing da semplice a terminale o da terminale a semplice.
 - Per un alias con una strategia di routing semplice, modifica l'ID della flotta a cui è associato l'alias.
 - Per un alias con una strategia di routing dei terminali, modifica il testo del messaggio.
3. Scegli Save changes (Salva modifiche). Quando aggiorni l'ID della flotta per un alias con una semplice strategia di routing, potrebbero essere necessari fino a 2 minuti per completare la transizione. Durante questo periodo, potrebbero verificarsi nuovi posizionamenti nelle sessioni di gioco sulla vecchia flotta.

AWS CLI

Utilizzate il [update-alias](#) comando per apportare modifiche a una risorsa alias. È possibile aggiornare una risorsa alias con l'impostazione predefinita Regione AWS corrente oppure aggiungere un `--region` tag per specificarne uno diverso. Regione AWS

È possibile modificare le seguenti proprietà:

- Nome alias.
- Descrizione dell'alias.
- Tipo di strategia di routing. Assicurati di fornire un ID della flotta o una stringa di messaggi per la nuova strategia di routing.
- Fleet ID per una strategia di routing semplice esistente. L'ID della flotta deve trovarsi nella stessa regione dell'alias.

- Stringa di messaggio per una strategia di routing dei terminali esistente.

Code delle sessioni di gioco nella console Amazon GameLift Servers

La risorsa queue rappresenta la configurazione del componente di posizionamento della sessione di gioco. Determina il modo in cui Amazon GameLift Servers cerca e sceglie la migliore risorsa di calcolo possibile per ospitare una nuova sessione di gioco, in base alle tue scelte di configurazione.

Visualizza le informazioni sulle code delle sessioni di gioco nella Amazon GameLift Servers console o utilizzando l'SDK o l'SDK per. AWS Amazon GameLift Servers

Console

Puoi visualizzare le informazioni su tutte le code, utilizzate per elaborare le richieste di nuove sessioni di gioco. La pagina delle code mostra le code delle sessioni di gioco nella zona attualmente selezionata. Regione AWS Nella pagina Code, è possibile creare una nuova coda, eliminare le code esistenti oppure aprire una pagina di dettaglio per una coda selezionata. Ogni pagina di dettaglio della coda contiene i dati dei parametri e della configurazione della coda. Per ulteriori informazioni sulle code, consulta [Configura il posizionamento delle sessioni di gioco](#).

La pagina delle code mostra le seguenti informazioni di riepilogo per ogni coda. È possibile modificare il contenuto della tabella in base alle esigenze utilizzando lo strumento Preferenze (vedere



icona

nell'angolo in alto a destra della tabella). Le preferenze personalizzate vengono salvate nell'utente AWS del tuo account e vengono applicate automaticamente ogni volta che visualizzi questa pagina.

- Nome coda: il nome assegnato alla coda. Le richieste di nuove sessioni di gioco specificano una coda utilizzando questo nome.
- Timeout della coda: periodo massimo, in secondi, durante il quale una richiesta di posizionamento della sessione di gioco rimane in coda prima del timeout.
- Destinazioni in coda: numero di flotte elencate nella configurazione della coda. Amazon GameLift Servers inserisce nuove sessioni di gioco su qualsiasi flotta in coda.

Visualizza i dettagli della coda

Puoi accedere a informazioni dettagliate su qualsiasi coda, inclusi la configurazione e i parametri della coda. Per aprire una pagina dei dettagli della coda, vai alla pagina Code e scegli un nome per la coda.

Nella pagina dei dettagli della coda è visualizzata una tabella di riepilogo e schede con informazioni aggiuntive. In questa pagina puoi eseguire le seguenti operazioni:

- Aggiorna la configurazione della coda, l'elenco delle destinazioni e le policy di latenza dei giocatori. Scegli Modifica.
- Eliminare una coda. Dopo aver eliminato una coda, tutte le richieste di nuove sessioni di gioco che fanno riferimento al nome della coda avranno esito negativo. Scegli Elimina.

Note

Per ripristinare una coda eliminata, crea una nuova coda con il nome della coda eliminata.

Informazioni

La sezione Panoramica mostra il nome della risorsa Amazon (ARN) e il Timeout della coda. Puoi utilizzare l'ARN per fare riferimento alla coda in altre operazioni o aree di Amazon GameLift Servers. Il timeout è l'intervallo di tempo massimo, in secondi, in cui la richiesta di posizionamento di una sessione di gioco resta nella coda prima che si verifichi un timeout.

Notifica degli eventi

La sezione Notifica degli eventi elenca l'argomento SNS in cui vengono Amazon GameLift Servers pubblicate le notifiche degli eventi e i dati sugli eventi che vengono aggiunti a tutti gli eventi creati da questa coda.

Tag

La tabella Tag mostra le chiavi e i valori utilizzati per etichettare la risorsa. Per ulteriori informazioni sull'etichettatura, consulta [Etichettatura delle risorse AWS](#).

Metriche

Nella scheda Parametri è mostrata una rappresentazione grafica dei parametri della coda nel tempo.

I parametri delle code includono una serie di informazioni che descrivono l'attività di posizionamento nella coda, inclusi i posizionamenti riusciti organizzati per regione. Puoi utilizzare i dati della regione per capire dove stai ospitando i tuoi giochi. Le metriche di posizionamento a livello regionale possono aiutare a individuare problemi relativi alla progettazione generale delle code.

Le metriche della coda sono disponibili anche in Amazon CloudWatch. Per le descrizioni delle metriche disponibili, consulta [Parametri Amazon GameLift Servers per code](#).

Destinazioni

La scheda Destinazioni mostra tutti i parchi istanze o gli alias elencati per la coda.

Quando Amazon GameLift Servers cerca nelle destinazioni le risorse disponibili per ospitare una nuova sessione di gioco, cerca nell'ordine predefinito elencato qui. Finché la prima destinazione in elenco dispone ancora di capacità, Amazon GameLift Servers vi posiziona le nuove sessioni di gioco. Puoi fare in modo che le singole richieste di posizionamento delle sessioni di gioco abbiano la priorità sull'ordine predefinito fornendo i dati di latenza dei giocatori. Tali dati indicano ad Amazon GameLift Servers di effettuare la ricerca di una destinazione disponibile con la latenza media dei giocatori più bassa. Per ulteriori informazioni sulla progettazione delle code, consulta [Personalizza una coda di sessioni di gioco](#).

Posizionamento delle sessioni

Impostazioni di configurazione che personalizzano il processo di posizionamento del gioco.

Policy di latenza dei giocatori

La sezione Policy di latenza dei giocatori mostra tutte le policy utilizzate dalla coda. Le tabelle elencano le politiche nell'ordine in cui vengono applicate.

Posizioni

La sezione Posizioni mostra le posizioni in cui questa coda può inserire una sessione di gioco.

Priorità

La sezione Priorità mostra l'ordine in cui la coda valuta i dettagli delle sessioni di gioco.

Ordine posizione

La sezione Ordine posizione mostra l'ordine predefinito utilizzato dalla coda per posizionare le sessioni di gioco. La coda utilizza questo ordine se non sono stati definiti altri tipi di priorità.

AWS SDK

Utilizza i seguenti comandi AWS CLI per recuperare informazioni su questa risorsa:

- [DescribeGameSessionQueues](#)

Preparati al lancio con l'Amazon GameLift Servershosting

Usa le seguenti liste di controllo per convalidare ogni fase di implementazione del gioco. Gli elementi contrassegnati come [Critici] sono fondamentali per il lancio della produzione.

[Scarica e completa il questionario di Amazon GameLift Servers lancio, disponibile nella Amazon GameLift Servers console.](#) Desideriamo che ogni sviluppatore di giochi abbia una giornata di lancio senza intoppi e le informazioni richieste ci aiutano a prepararti per i prossimi test di carico, il soft launch o il lancio pubblico. Amazon GameLift Servers Pianifica di inviare il questionario compilato almeno tre (3) mesi prima del primo test di carico.

Argomenti

- [Prepara il gioco](#)
- [Preparati per il test](#)
- [Preparati per il lancio](#)
- [Pianifica gli aggiornamenti post-lancio](#)

Prepara il gioco

- [Critico] Verifica di aver completato tutti i [passaggi della roadmap di sviluppo](#) della tua soluzione di hosting e di disporre di tutti i componenti necessari, tra cui un server di gioco integrato, un servizio di backend per client di gioco, flotte di hosting e un metodo di posizionamento delle sessioni di gioco (ad esempio una coda).
- [Critico] [Crea ruoli AWS Identity and Access Management \(IAM\)](#) che consentano al server di gioco di accedere ad altre AWS risorse durante l'esecuzione.
- [Critico] Progetta e implementa il failover su altre risorse di hosting in base alle esigenze.
- [Pianifica l'implementazione delle flotte verso le località di destinazione](#), considerando la coda di gioco e la struttura della flotta.

- [Automatizza la distribuzione](#) utilizzando Infrastructure as Code (IaC) con e il CloudFormation AWS Cloud Development Kit (AWS CDK)
- [Raccogli log e analisi](#) utilizzando Amazon CloudWatch e Amazon Simple Storage Service (Amazon S3).

Preparati per il test

- [Critico] [Richiedi aumenti per le quote di Amazon GameLift Servers servizio](#) e altre Servizio AWS quote in modo che il tuo ambiente live possa adattarsi alle esigenze di produzione.
- [Critico] Verificate che le porte aperte sulle flotte attive corrispondano alla gamma di porte che i server potrebbero utilizzare.
- [Critico] Chiudi la porta RDP 3389 e la porta SSH 22.
- Sviluppa un piano per la DevOps gestione del gioco. Se utilizzi Amazon CloudWatch Logs o i parametri CloudWatch personalizzati di Amazon, definisci gli allarmi per problemi gravi o critici sulla flotta di server. Simula gli errori e testa i runbook.
- Verifica che le risorse di elaborazione che stai utilizzando siano in grado di supportare il numero di processi server che desideri eseguire contemporaneamente su ogni elaborazione.
- [Ottimizza la tua politica di scalabilità](#) in modo che sia inizialmente più conservativa e fornisca più capacità inattiva di quella che ritieni necessaria. Puoi ottimizzare i costi in un secondo momento. Prendi in considerazione l'utilizzo di una politica di scalabilità basata sugli obiettivi con una capacità inattiva del 20%.
- Ad esempio FlexMatch, usa [le regole di latenza](#) per abbinare giocatori geograficamente vicini l'uno all'altro. Verifica come si comporta sotto carico con i dati di latenza sintetici del tuo client di test di carico.
- Esegui il test di caricamento dell'infrastruttura di autenticazione dei giocatori e delle sessioni di gioco per vedere se è scalabile in modo efficace per soddisfare la domanda.
- Verifica che un server rimasto in funzione per diversi giorni possa ancora accettare connessioni.
- Innalza il livello del Supporto piano a Business o Enterprise in modo che AWS possa risponderti in caso di problemi o interruzioni.

Preparati per il lancio

- [Critico] [Imposta la politica di protezione della flotta](#) sulla protezione completa di tutte le flotte attive in modo che il ridimensionamento non interrompa le sessioni di gioco attive.

- [Critico] [Imposta le dimensioni massime della flotta](#) sufficientemente elevate da soddisfare al minimo i picchi di domanda previsti. Ti consigliamo di raddoppiare la dimensione massima in caso di domanda imprevista.
- Incoraggia l'intero team di sviluppo a partecipare all'evento di lancio e a monitorare il lancio del gioco in una sala lancio.
- Monitora la latenza e l'esperienza dei giocatori.

Pianifica gli aggiornamenti post-lancio

- [Ottimizza la politica di scalabilità](#) per ridurre al minimo la capacità inattiva in base all'utilizzo del giocatore.
- [Modifica FlexMatch le regole](#) o [aggiungi sedi di hosting in](#) base ai dati sulla latenza dei giocatori e ai requisiti rivisti.
- Ottimizza la configurazione di runtime per eseguire quante più sessioni di gioco possibile su ciascuna risorsa di elaborazione. Massimizzare l'efficienza delle prestazioni in questo modo può influire direttamente sui costi del parco macchine, perché potresti essere in grado di eseguire più processi server con le stesse risorse di elaborazione.
- [Usa i tuoi dati di analisi](#) per promuovere lo sviluppo continuo, migliorare l'esperienza dei giocatori e la longevità del gioco e ottimizzare la monetizzazione.

Gestisci le risorse di Amazon GameLift Servers hosting utilizzando CloudFormation

Puoi utilizzarlo CloudFormation per gestire le tue Amazon GameLift Servers risorse. In CloudFormation, crei un modello che modella ogni risorsa e poi usi il modello per creare le tue risorse. Per aggiornare le risorse, apporti le modifiche al modello e le utilizzi CloudFormation per implementare gli aggiornamenti. È possibile organizzare le risorse in gruppi logici, chiamati stack e set di stack.

L'utilizzo CloudFormation per mantenere le risorse di Amazon GameLift Servers hosting offre un modo più efficiente per gestire set di AWS risorse. Puoi utilizzare il controllo della versione per tenere traccia delle modifiche dei modelli nel corso tempo e coordinare gli aggiornamenti effettuati da più membri del team. È inoltre possibile riutilizzare i modelli. Ad esempio, quando distribuisce un gioco in più regioni, puoi utilizzare lo stesso modello per creare risorse identiche in ogni regione. È inoltre possibile utilizzare questi modelli per distribuire gli stessi set di risorse in un'altra partizione.

Per ulteriori informazioni in merito CloudFormation, consulta la [Guida AWS CloudFormation per l'utente](#). Per visualizzare le informazioni relative ai modelli di Amazon GameLift Servers risorse, consulta il [riferimento ai tipi di Amazon GameLift Servers risorse](#).

Best practice

Per una guida dettagliata sull'utilizzo CloudFormation, consulta le [CloudFormation migliori pratiche](#) nella Guida AWS CloudFormation per l'utente. Queste best practice hanno particolare rilevanza con Amazon GameLift Servers.

- Gestisci in modo coerente le tue risorse tramite CloudFormation. Se modifichi le tue risorse al di fuori delle CloudFormation tue risorse, le tue risorse non saranno sincronizzate con i tuoi modelli di risorse.
- Usa CloudFormation pile e set di stack per gestire in modo efficiente più risorse.
 - Usa gli stack per gestire gruppi di risorse connesse. Ad esempio, uno stack che contiene una build, una flotta che fa riferimento alla build e un alias che fa riferimento alla flotta. Se aggiorni il modello per sostituire una build, CloudFormation sostituisce le flotte collegate alla build. CloudFormation quindi aggiorna gli alias esistenti in modo che puntino alle nuove flotte. Per ulteriori informazioni, consulta [Working with stacks](#) nella Guida per l'AWS CloudFormation utente.
 - Usa i set di CloudFormation stack se stai distribuendo stack identici in più regioni o account. AWS Per ulteriori informazioni, consulta [Lavorare con i set di stack](#) nella Guida per l'utente AWS CloudFormation
- Se utilizzi istanze Spot, includi un parco istanze on demand come backup. Ti consigliamo di configurare i tuoi modelli con due flotte in ogni regione, un parco con istanze Spot e un parco con istanze On-Demand.
- Raggruppa le risorse specifiche della località e le risorse globali in pile separate quando gestisci risorse in più sedi.
- Posiziona le tue risorse globali vicino ai servizi che le utilizzano. Risorse come le code e le configurazioni di matchmaking tendono a ricevere un volume elevato di richieste da fonti specifiche. Posizionando le risorse vicino alla fonte di tali richieste, riduci al minimo i tempi di viaggio delle richieste e puoi migliorare le prestazioni complessive.
- Posiziona la configurazione dell'abbinamento nella stessa regione della coda delle sessioni di gioco utilizzata.
- Crea un alias separato per ogni parco istanze nello stack.

Utilizzo delle CloudFormation pile

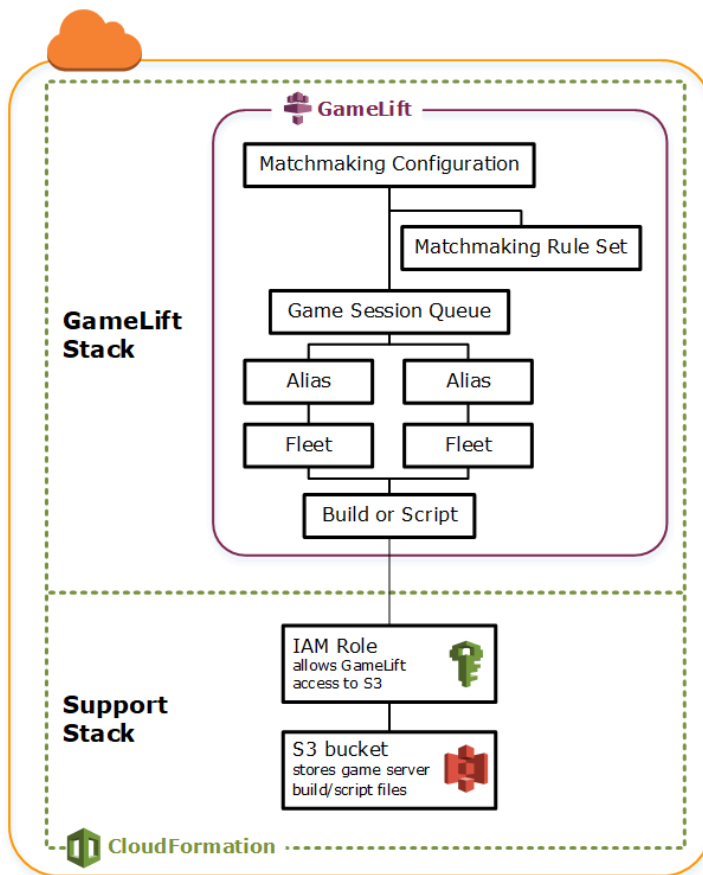
Consigliamo di utilizzare le seguenti strutture per configurare gli CloudFormation stack per le Amazon GameLift Servers risorse. La struttura ottimale dello stack varia a seconda che il gioco venga distribuito in una o più postazioni.

Pile per una singola postazione

Per gestire Amazon GameLift Servers le risorse in un'unica posizione, consigliamo una struttura a due stack:

- **Support stack:** questo stack contiene risorse da cui dipendono Amazon GameLift Servers le tue risorse. Come minimo, questo stack deve includere il bucket S3 in cui si memorizzano i file di script Realtime o il server di gioco personalizzato. Lo stack dovrebbe includere anche un ruolo IAM che Amazon GameLift Servers autorizzi a recuperare i file dal bucket S3 durante la creazione di una risorsa di build o script. Amazon GameLift Servers Questo stack potrebbe contenere anche altre AWS risorse utilizzate con il gioco, come tabelle DynamoDB, cluster Amazon Redshift e funzioni Lambda.
- **Amazon GameLift Serversstack:** questo stack contiene tutte le tue Amazon GameLift Servers risorse, inclusa la build o lo script, un set di flotte, alias e la coda delle sessioni di gioco. CloudFormation crea una risorsa di compilazione o script con file archiviati nella posizione del bucket S3 e distribuisce la build o lo script su una o più risorse del parco risorse. Ogni parco istanze deve avere un alias corrispondente. La coda delle sessioni di gioco fa riferimento ad alcuni o tutti gli alias del parco istanze. Se lo utilizzi FlexMatch per il matchmaking, questo stack contiene anche una configurazione di matchmaking e un set di regole.

Il diagramma seguente illustra una struttura a due livelli per la distribuzione delle risorse in una singola regione. AWS



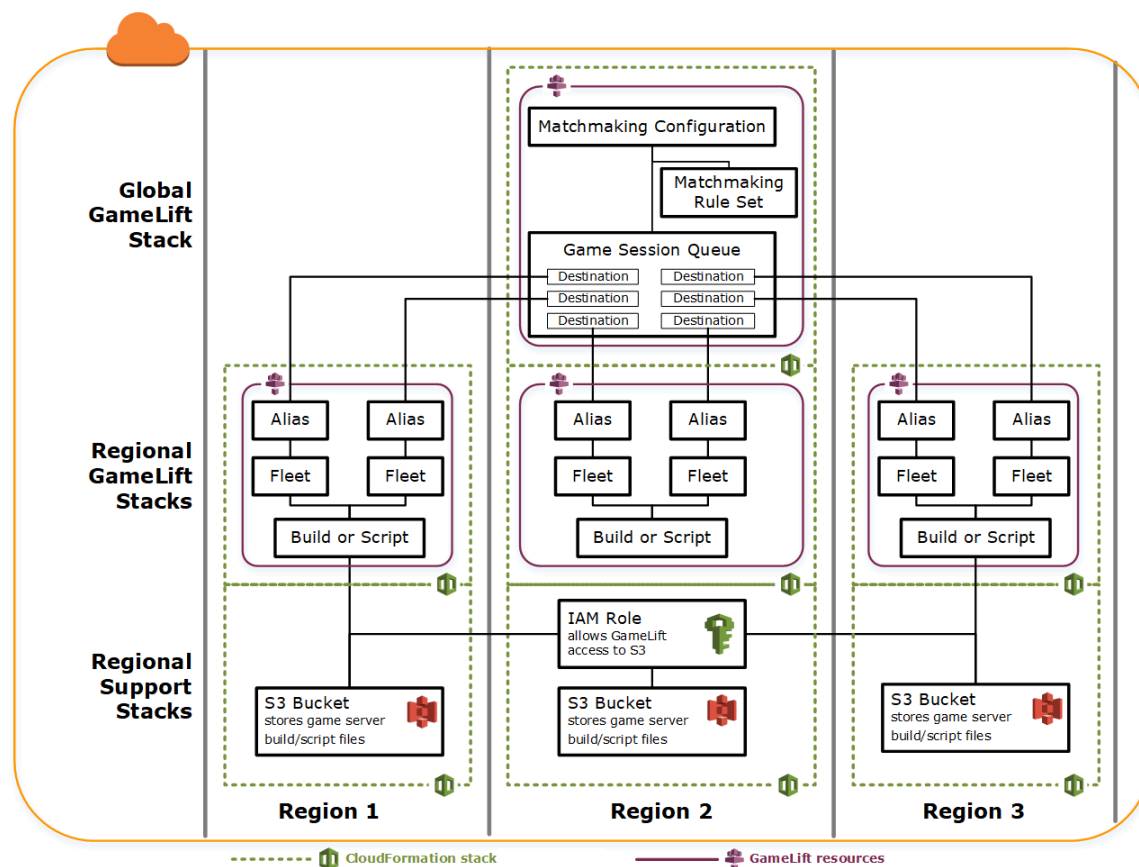
Pile per più regioni

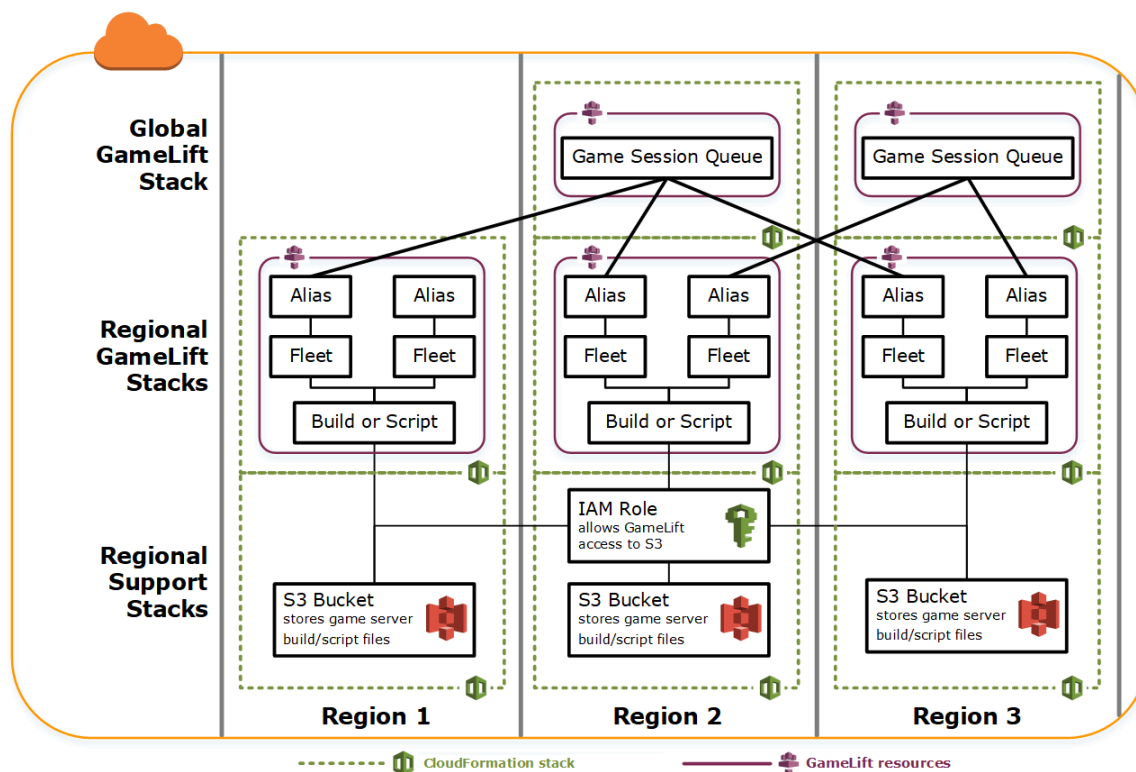
Quando distribuisce il tuo gioco in più di una regione, tieni presente come le risorse possono interagire tra le regioni. Alcune risorse, come i parchi istanze Amazon GameLift Servers, possono fare riferimento solo ad altre risorse nella stessa regione. Altre risorse, come una Amazon GameLift Servers coda, sono indipendenti dalla regione. Per gestire le risorse Amazon GameLift Servers in più regioni, consigliamo la seguente struttura.

- Stack di supporto regionali: questi stack contengono risorse da cui dipendono le tue risorse.
 - Amazon GameLift Servers Questo stack deve includere il bucket S3 in cui si memorizzano il server di gioco personalizzato o i file di script Realtime. Potrebbe contenere anche altre AWS risorse per il tuo gioco, come tabelle DynamoDB, cluster Amazon Redshift e funzioni Lambda. Molte di queste risorse sono specifiche per regione, quindi è necessario crearle in ogni regione. Amazon GameLift Servers necessita inoltre di un ruolo IAM che consenta l'accesso a queste risorse di supporto. Poiché un ruolo IAM è indipendente dalla regione, è necessaria una sola risorsa di ruolo, collocata in ogni regione e referenziata in tutti gli altri stack di supporto.

- **Amazon GameLift ServersStack regionali:** questo stack contiene le Amazon GameLift Servers risorse che devono esistere in ogni regione in cui viene distribuito il gioco, tra cui la build o lo script, un set di flotte e alias. CloudFormation crea una risorsa di compilazione o script con file in una posizione del bucket S3 e distribuisce la build o lo script su una o più risorse del parco risorse. Ogni parco istanze deve avere un alias corrispondente. La coda delle sessioni di gioco fa riferimento ad alcuni o tutti gli alias del parco istanze. Puoi mantenere un modello per descrivere questo tipo di stack e utilizzarlo per creare set di risorse identici in ogni regione.
- **Stack globale:** questo Amazon GameLift Servers stack contiene la coda della sessione di gioco e le risorse di matchmaking. Queste risorse possono essere localizzate in qualsiasi regione e di solito sono posizionate nella stessa regione. La coda può fare riferimento ai parchi istanze o agli alias che si trovano in qualsiasi regione. Per posizionare code aggiuntive in regioni diverse, crea altri stack globali.

I diagrammi seguenti illustrano una struttura multistack per la distribuzione di risorse in diverse regioni. AWS Il primo diagramma mostra una struttura per una singola coda di sessioni di gioco. Il secondo diagramma mostra una struttura con più code.





Aggiornamento delle build

Le build Amazon GameLift Servers sono immutabili, così come la relazione tra una build e un parco istanze. Di conseguenza, quando aggiorni le risorse di hosting per utilizzare un nuovo set di file della build di gioco, è necessario effettuare quanto segue:

- Creare una nuova build utilizzando il nuovo set di file (sostituzione).
- Creare un nuovo set di parchi istanze per distribuire la nuova build del gioco (sostituzione).
- Reindirizzare gli alias per puntare ai nuovi parchi istanze (aggiornamento senza interruzioni).

Per ulteriori informazioni, consulta [Aggiornare i comportamenti delle risorse dello stack](#) nella Guida per l'AWS CloudFormation utente.

Distribuisce automaticamente gli aggiornamenti delle build

Quando si aggiorna uno stack contenente le relative risorse di build, fleet e alias, il CloudFormation comportamento predefinito prevede l'esecuzione automatica di questi passaggi in sequenza. Attivi questo aggiornamento caricando prima i nuovi file della build in un nuovo percorso S3. Quindi modifichi il modello di CloudFormation build in modo che punti alla nuova posizione S3. Quando aggiorni lo stack con il nuovo percorso S3, viene attivata la seguente sequenza CloudFormation :

1. Recupera i nuovi file da S3, convalida i file e crea una nuova build Amazon GameLift Servers.
2. Aggiorna il riferimento della build nel modello del parco istanze che attiva la creazione di nuovi parchi istanze.
3. Una volta che i nuovi parchi istanze sono attivi, aggiorna il riferimento del parco istanze nell'alias che attiva l'aggiornamento dell'alias per indirizzare i nuovi parchi istanze.
4. Elimina il vecchio parco istanze.
5. Elimina la vecchia build.

Se la coda delle sessioni di gioco utilizza gli alias del parco istanze, il traffico dei giocatori viene automaticamente trasferito ai nuovi parchi istanze non appena gli alias vengono aggiornati. I vecchi parchi istanze vengono gradualmente svuotati dei giocatori alla fine delle sessioni di gioco. La scalabilità automatica si occupa dell'aggiunta e della rimozione di istanze da ogni set di flotte in base alle fluttuazioni del traffico dei giocatori. In alternativa, è possibile specificare un numero iniziale di istanze desiderato per aumentare rapidamente lo switch e abilitare la scalabilità automatica in un secondo momento.

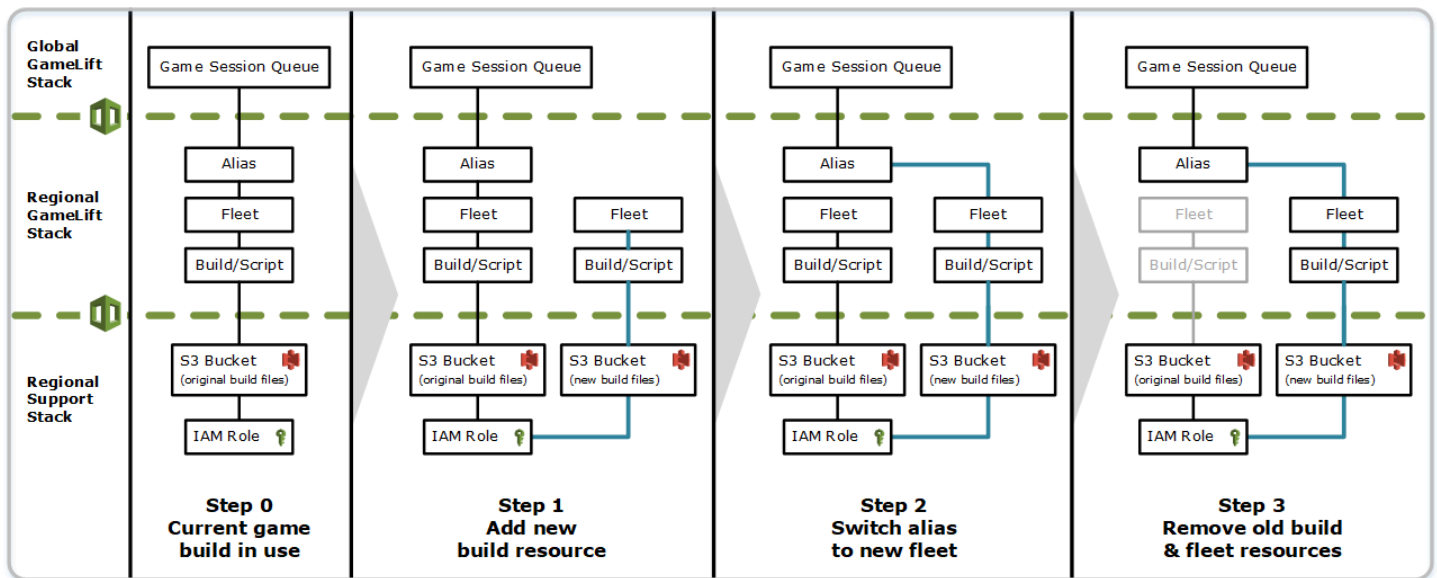
È inoltre possibile impostare le risorse di CloudFormation conservazione anziché eliminarle. Per ulteriori informazioni, consulta [RetainResources](#) nella documentazione di riferimento dell'API AWS CloudFormation .

Distribuisci gli aggiornamenti delle build manualmente

Se vuoi avere maggiore controllo su quando i nuovi parchi istanze sono disponibili per i giocatori, hai diverse opzioni. Puoi scegliere di gestire manualmente gli alias utilizzando la console Amazon GameLift Servers o l'interfaccia a riga di comando. In alternativa, invece di aggiornare il modello di build per sostituire la build e i parchi istanze, è possibile aggiungere al modello un secondo set di definizioni di build e parco istanze. Quando aggiorni il modello, CloudFormation crea una seconda risorsa di compilazione e le flotte corrispondenti. Poiché le risorse esistenti non vengono sostituite, non vengono cancellate e gli alias continuano a puntare i parchi istanze originali.

Il vantaggio principale di questo approccio è che ti offre la flessibilità. Puoi creare risorse separate per la nuova versione della tua build, testare le nuove risorse e controllare quando i nuovi parchi istanze diventano disponibili per i giocatori. Un potenziale svantaggio è che richiede il doppio delle risorse in ciascuna regione per un breve periodo di tempo.

Il diagramma seguente illustra tale processo.



Come funzionano i rollback

Quando si esegue un aggiornamento delle risorse, se un passaggio non viene completato, CloudFormation avvia automaticamente un rollback. Questo processo ripristina ogni passaggio della sequenza, eliminando le risorse appena create.

Se è necessario attivare manualmente un rollback, modifica la chiave di percorso S3 del modello di build sul percorso originale e aggiorna lo stack. Viene creata una nuova build Amazon GameLift Servers e un nuovo parco istanze e l'alias passa al nuovo parco istanze una volta che diventa attivo. Se gestisci gli alias separatamente, devi modificarli per puntare ai nuovi parchi istanze.

Per ulteriori informazioni su come gestire un rollback che fallisce o si blocca, consulta [Continuare a ripristinare un aggiornamento nella Guida](#) per l'AWS CloudFormation utente.

DDoS Protezione S con Amazon GameLift Servers

Amazon GameLift Servers player gateway è una rete basata su relè che indirizza il traffico UDP tra client di gioco e server di gioco ospitati su Amazon GameLift Servers Player gateway offre una protezione DDoS proattiva convalidando il traffico prima che raggiunga i server di gioco, limitando la velocità del traffico dei giocatori, nascondendo gli indirizzi IP dei server di gioco al pubblico e fornendo endpoint aggiornati quando gli endpoint di inoltro non funzionano correttamente. Questa funzionalità aiuta a proteggere i server di gioco dagli attacchi volumetrici con una latenza aggiuntiva trascurabile.

Player Gateway richiede che i server di gioco funzionino su flotte EC2 gestite o flotte di container basate su Linux Amazon GameLift Servers. Devi abilitare il player gateway durante la creazione della flotta e aggiornare il client di gioco e il backend di gioco per utilizzare questa funzione.

Vantaggi principali

Player gateway offre i seguenti vantaggi operativi e di sicurezza:

- Nascondi gli indirizzi IP dei server di gioco: i client di gioco si connettono tramite endpoint di inoltro anziché direttamente ai server di gioco, nascondendo gli indirizzi dei server di gioco al pubblico.
- Convalida il traffico: tutto il traffico attraverso il gateway dei giocatori richiede un token di gateway del giocatore, che consente solo al traffico proveniente da giocatori autenticati di raggiungere i tuoi server di gioco.
- Sostituzione dinamica degli endpoint: quando gli endpoint di inoltro non funzionano correttamente, sostituisce Amazon GameLift Servers rapidamente gli endpoint e fornisce endpoint integri aggiornati alla chiamata successiva per aggiornare i dettagli di connessione del giocatore.
- Distribuisci il traffico dei giocatori: gli endpoint di inoltro variano tra i giocatori, riducendo l'impatto di un singolo endpoint di inoltro non funzionante sugli altri giocatori all'interno della stessa sessione di gioco.

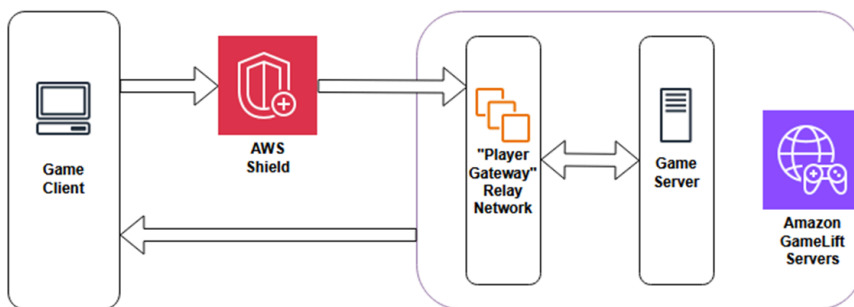
Come funziona Player Gateway

Il gateway del giocatore utilizza una rete di inoltro per instradare il traffico UDP tra client di gioco e server di gioco. Ciò protegge dagli attacchi DDoS convalidando il traffico prima che raggiunga i server di gioco, limitando la velocità del traffico dei giocatori, nascondendo gli indirizzi IP dei server

di gioco al pubblico e fornendo endpoint aggiornati quando gli endpoint di inoltro non funzionano correttamente.

Flusso di traffico

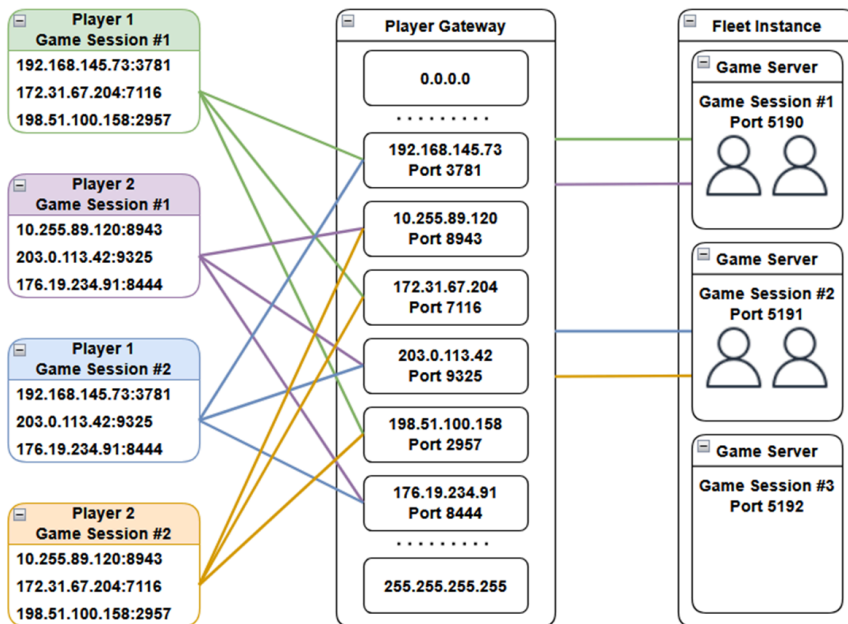
Quando un giocatore si connette a una sessione di gioco, il backend di gioco recupera gli endpoint di inoltro e un token gateway del giocatore dall'API e li invia al client di gioco. `GetPlayerConnectionDetails` Il client di gioco aggiunge il token gateway del giocatore ai pacchetti UDP e invia i pacchetti a un endpoint di inoltro. La rete di inoltro convalida il token e indirizza il traffico legittimo al server di gioco. Prima della consegna, la rete di inoltro rimuove il token gateway del giocatore in modo che i server di gioco ricevano il payload non elaborato del client di gioco e non richiedano modifiche al codice. La comunicazione dal server di gioco al client di gioco ritorna attraverso la rete di inoltro senza modifiche.



Concetti principali

Endpoint del relè

Gli endpoint di inoltro sono combinazioni di indirizzi IP e porte utilizzate dai client di gioco per inviare traffico attraverso il gateway del giocatore. Ogni giocatore riceve più endpoint che variano da giocatore a giocatore per distribuire il traffico e ridurre l'impatto degli attacchi sugli altri giocatori nella stessa sessione di gioco.



Token gateway per i giocatori

I token Player Gateway sono array di byte crittografati che autorizzano un giocatore a inviare traffico a una sessione di gioco tramite il player gateway. L'GetPlayerConnectionDetailsAPI restituisce i token come stringhe con codifica base64. I client di gioco devono aggiungere il token gateway del giocatore a ogni pacchetto UDP. La rete di inoltro convalida i token e rifiuta i pacchetti con token non validi o mancanti.

Important

I token gateway del giocatore non devono essere crittografati e devono apparire all'inizio di ogni pacchetto UDP inviato dal client di gioco. Se il gioco crittografa i payload, aggiungi il token gateway del giocatore non modificato dopo aver crittografato i dati di gioco.

I token Player Gateway rimangono validi per almeno 3 minuti. Consigliamo di aggiornare i token ogni 60 secondi per garantire che i giocatori ricevano endpoint aggiornati quando gli endpoint di inoltro non funzionano correttamente.

GetPlayerConnectionDetails API

Il backend di gioco chiama l'GetPlayerConnectionDetailsAPI per recuperare i dettagli di connessione per i giocatori in una sessione di gioco. L'API restituisce gli endpoint di inoltro e i token

gateway del giocatore oppure ricorre all'indirizzo IP e alla porta del server di gioco per la connessione diretta. Il client di gioco deve essere progettato per gestire entrambi i tipi di connessione. Per ricevere endpoint aggiornati quando gli endpoint di inoltro non funzionano correttamente, chiama questa API periodicamente durante la sessione di gioco (consigliato ogni 60 secondi).

Per ulteriori informazioni, consulta [GetPlayerConnectionDetails API](#).

Monitoraggio del gateway dei giocatori

Player gateway pubblica metriche su Amazon CloudWatch per aiutarti a monitorare i modelli di traffico di rete, identificare potenziali attacchi DDoS e monitorare le prestazioni dei relay. Le metriche includono pacchetti e byte in entrata/uscita, traffico limitato e sessioni di gioco. Per l'elenco completo delle metriche relative al player gateway, consulta [DDoS Metriche di protezione S \(player gateway\)](#).

IPv4 e compatibilità IPv6

I client di gioco comunicano utilizzando IPv4. Player Gateway utilizza IPv6 per comunicare con i server di gioco. Amazon GameLift Servers gestisce automaticamente la traduzione tra IPv4 e IPv6 in base alla configurazione della flotta.

Per ulteriori informazioni sulla configurazione del player gateway sulla tua flotta, consulta [CreateFleet](#) o [CreateContainerFleet](#). Per ulteriori informazioni IPv4 e IPv6 assistenza, consulta [IPv4 e IPv6 compatibilità](#).

Abilita il gateway per i giocatori sulle flotte

Attiva il player gateway durante la creazione di una flotta per indirizzare il traffico di gioco attraverso endpoint di inoltro che convalidano e limitano la velocità del traffico UDP in entrata verso i tuoi server di gioco.

Prerequisiti

Prima di abilitare Player Gateway, verifica i seguenti requisiti:

- Località supportate: per un elenco delle AWS località che supportano il player gateway, vedi [Amazon GameLift Servers sedi di assistenza](#).
- Tipo di flotta: il gateway Player supporta flotte Managed EC2 e flotte Managed Container basate su Linux.

- **Server SDK:** il gateway Player richiede Server SDK 5.0 o versione successiva. Amazon GameLift Servers

Modalità Player Gateway

Configura la compatibilità del player gateway per la tua flotta utilizzando la `PlayerGatewayMode` proprietà:

DISABILITATO (impostazione predefinita)

Fleet non è compatibile con Player Gateway.

ABILITATO

Fleet è compatibile con il player gateway nelle località in cui è supportato. Le sedi della flotta che non supportano questa funzionalità possono comunque essere aggiunte alla flotta, ma non utilizzeranno il gateway del giocatore, utilizzando direttamente l'indirizzo IP del server di gioco.

REQUIRED

Fleet è compatibile con il gateway del giocatore. Le sedi della flotta che non supportano questa funzionalità non possono essere aggiunte al parco veicoli.

IPv4 e IPv6 compatibilità

I client di gioco devono comunicare con il gateway del giocatore tramite IPv4. La rete Player Gateway Relay comunica con i server di gioco tramite IPv6. Le flotte di container gestite convertono automaticamente il traffico IPv6 in IPv4. Per le flotte gestite da EC2, configura il modo in cui la flotta gestisce il traffico utilizzando la proprietà in: `IPv6 GameServerIpProtocolSupported PlayerGatewayConfiguration`

IPv4 (impostazione predefinita)

Il tuo server di gioco accetta solo IPv4 traffico in entrata. Sulle istanze verrà installato ed eseguito un leggero software di traduzione IP per ricevere e trasformare il traffico in entrata IPv6 . IPv4

DUAL_STACK

Il tuo server di gioco è compatibile con il traffico in entrata. IPv6 Il software di traduzione IP non verrà installato o eseguito sulle tue istanze. I server di gioco che supportano il supporto IPv6 nativo potrebbero avere prestazioni leggermente superiori.

Integra il gateway del giocatore in un gioco

Dopo aver creato una flotta di giocatori abilitata al gateway per i giocatori, devi aggiornare il client di gioco e il backend di gioco per integrarli con Player Gateway. Il backend di gioco recupera gli endpoint relay e i token del gateway del giocatore, quindi li fornisce ai client di gioco. I client di gioco inviano tutto il traffico UDP a questi endpoint di inoltro.

L'integrazione segue questo percorso di chiamata:

1. Il backend di gioco chiama l'`GetPlayerConnectionDetailsAPI` per recuperare gli endpoint di inoltro e i token gateway del giocatore per ogni giocatore in una sessione di gioco.
2. Il backend di gioco invia gli endpoint relay e i token gateway del giocatore al client di gioco.
3. Il client di gioco aggiunge il token gateway del giocatore a tutti i pacchetti UDP in uscita e invia i pacchetti agli endpoint di inoltro.
4. La rete di inoltro convalida il token gateway del giocatore e indirizza il traffico verso il server di gioco.
5. Il server di gioco invia il traffico al client di gioco attraverso gli stessi percorsi di inoltro.

Integrazione con il backend

Il backend del gioco deve chiamare l'`GetPlayerConnectionDetailsAPI` per recuperare gli endpoint di inoltro e i token di accesso ai giocatori. Il backend fornisce quindi queste informazioni ai client di gioco.

GetPlayerConnectionDetails API

L'`GetPlayerConnectionDetailsAPI` restituisce i dettagli di connessione per i giocatori in una sessione di gioco:

Endpoint e token di inoltro

Quando il player gateway è abilitato e supportato nella località, restituisce:

- Endpoint di inoltro: più endpoint di inoltro (indirizzo IP e porta) che variano a seconda dei giocatori
- Token gateway Player: token che il client deve anteporre a tutti i pacchetti UDP (valido per almeno 3 minuti)
- Scadenza: orario di scadenza del token Player Gateway

Connessione diretta

Quando il gateway del giocatore non è abilitato o non è supportato nella località, restituisce l'indirizzo IP e la porta del server di gioco

Il client di gioco deve essere progettato per gestire entrambi i tipi di connessione.

Esempio di chiamata API:

```
// C++ example using AWS SDK
Aws::GameLift::GameLiftClient client;
Aws::GameLift::Model::GetPlayerConnectionDetailsRequest request;
request.SetGameSessionId(gameSessionId);
request.SetPlayerIds(playerIds); // Vector of player IDs

auto outcome = client.GetPlayerConnectionDetails(request);
if (outcome.IsSuccess()) {
    auto result = outcome.GetResult();
    auto connectionDetails = result.GetPlayerConnectionDetails();

    // Process each player's connection details
    for (const auto& detail : connectionDetails) {
        std::string playerId = detail.GetPlayerId();

        // Get relay endpoints (IP address and port)
        auto endpoints = detail.GetEndpoints();
        for (const auto& endpoint : endpoints) {
            std::string ipAddress = endpoint.GetIpAddress();
            int port = endpoint.GetPort();
        }

        // Get player gateway token
        auto token = detail.GetPlayerGatewayToken();

        // Get expiration time
        auto expiration = detail.GetExpiration();

        // Send endpoints and token to game client
    }
}
```

Vedi [GetPlayerConnectionDetails](#) nell'Amazon GameLift Servers API Reference.

Aggiorna gli endpoint del relè

Chiama `GetPlayerConnectionDetails` periodicamente per fornire ai giocatori gli endpoint aggiornati quando gli endpoint delle staffette non funzionano correttamente. Consigliamo di chiamare l'API ogni 60 secondi per tutti i giocatori in una sessione di gioco. Per ottimizzare l'utilizzo dell'API, raggruppa più giocatori IDs in un'unica chiamata API.

Important

Le chiamate di aggiornamento regolari sono il meccanismo principale per fornire endpoint aggiornati ai giocatori. Sebbene i token gateway dei giocatori rimangano validi per almeno 3 minuti, l'aggiornamento ogni 60 secondi garantisce ai giocatori di ricevere regolarmente gli endpoint aggiornati.

Integrazione con i client

Il client di gioco deve anteporre i token del gateway del giocatore a tutti i pacchetti UDP in uscita e inviare i pacchetti agli endpoint di inoltro forniti. Il Amazon GameLift Servers client SDK for C++ e Unreal Engine fornisce utilità per semplificare questa integrazione.

Requisiti del cliente

Per indirizzare il traffico attraverso il gateway del giocatore, il client di gioco deve:

- **Anteponi i token del gateway del giocatore:** sostituisci il token gateway del giocatore a tutti i pacchetti UDP in uscita. I token gateway del giocatore non devono essere crittografati e devono apparire all'inizio di ogni pacchetto. I pacchetti senza un token gateway per il giocatore valido verranno eliminati.
- **Invia agli endpoint di inoltro:** invia tutti i pacchetti UDP agli endpoint di inoltro forniti.
- **Mantieni l'attività di connessione:** assicurati che il client di gioco invii un pacchetto al server di gioco o che il server di gioco invii un pacchetto al giocatore, almeno una volta ogni 30 secondi. Questa attività mantiene la connessione attraverso la rete di inoltro.
- **Gestisci gli aggiornamenti degli endpoint:** accetta endpoint relay e token player gateway aggiornati dal tuo backend (consigliato ogni 60 secondi) e passa a nuovi endpoint senza interrompere la connessione.

Client SDK

Il Amazon GameLift Servers client SDK for C++ e Unreal Engine fornisce utilità per semplificare l'integrazione del player gateway:

- Gestione dei token: aggiunge i token gateway del giocatore a tutti i pacchetti UDP in uscita.
- Selezione degli endpoint: indirizza il traffico verso gli endpoint di inoltro utilizzando un algoritmo configurabile.
- Aggiornamento degli endpoint: pianifica callback periodici per recuperare gli endpoint di inoltro e i token del player gateway aggiornati dal backend.

Algoritmi di selezione degli endpoint

L'SDK del client fornisce due algoritmi integrati per la selezione dell'endpoint di inoltro da utilizzare:

Algoritmo di fallback

Utilizza un endpoint finché non diventa incompleto, quindi passa a un altro endpoint. Ideale per menu, lobby e giochi a turni in cui sono accettabili brevi interruzioni. Durante il failover degli endpoint, i pacchetti potrebbero andare persi per il periodo di timeout configurato (impostazione predefinita: 2 secondi).

Algoritmo di rotazione predittiva

Ruota continuamente tra tutti gli endpoint disponibili e prevede i guasti prima che si verifichino. Ideale per giochi in tempo reale come sparatutto in prima persona e giochi di corse in cui la consegna costante dei pacchetti è fondamentale. Richiede che il server di gioco invii messaggi con una frequenza costante.

Puoi anche implementare algoritmi personalizzati estendendo la classe di algoritmi base dell'SDK.

Risorse Client SDK

Per istruzioni complete sull'integrazione e codice di esempio, consulta le seguenti risorse:

C++

Un client SDK per C++ è [Amazon GameLift Servers disponibile presso client SDK for C++](#).

Unreal Engine

Un plug-in per Unreal Engine è disponibile presso il [Amazon GameLift Servers SDK](#) per Unreal Engine. L'accesso richiede l'iscrizione all'organizzazione Epic Games il. GitHub Vedi [Unreal Engine su GitHub per maggiori dettagli](#).

Metti alla prova la tua integrazione

Prima di passare a una flotta, verifica l'integrazione del player gateway a livello locale utilizzando lo strumento di test player gateway. Questo strumento simula l'infrastruttura proxy UDP del gateway del giocatore e ti aiuta a verificare che il tuo client di gioco inserisca correttamente i token del gateway del giocatore, instrada il traffico attraverso più endpoint e gestisca il degrado della rete.

[Per istruzioni di configurazione e utilizzo, consulta l'app Player Gateway Testing su GitHub](#)

Best practice

Segui queste best practice per integrare player gateway:

- Aggiorna gli endpoint ogni 60 secondi: chiama **GetPlayerConnectionDetails** ogni 60 secondi per assicurarti che i giocatori ricevano regolarmente gli endpoint aggiornati.
- Chiamate API in batch: quando chiamate **GetPlayerConnectionDetails** per più giocatori nella stessa sessione di gioco, raggruppa le chiamate in batch per ridurre il sovraccarico dell'API.
- Mantieni l'attività di connessione: assicurati che il traffico fluisca tra il client di gioco e il server di gioco almeno una volta ogni 30 secondi. Può trattarsi sia client-to-server del traffico che server-to-client del traffico.
- Gestisci gli aggiornamenti degli endpoint con garbo: quando il client di gioco riceve endpoint aggiornati dal backend, passa ai nuovi endpoint senza interrompere la connessione del giocatore.
- Scegli l'algoritmo giusto: seleziona l'algoritmo di selezione degli endpoint che meglio soddisfa i requisiti del tuo gioco. Usa l'algoritmo di fallback per le partite a turni e l'algoritmo di rotazione predittiva per le partite in tempo reale.

Test e risoluzione dei problemi Amazon GameLift Servers

Questa sezione fornisce indicazioni per la diagnosi e la risoluzione dei problemi con la tua soluzione di GameLift hosting Amazon. Che si verifichino problemi durante le operazioni di sviluppo, distribuzione o produzione, queste risorse ti aiuteranno a identificare le cause profonde e a implementare soluzioni efficaci.

- Ambienti di sviluppo e test: configura flussi di lavoro di sviluppo iterativi che rispecchiano le condizioni di produzione, consentendoti di individuare e risolvere i problemi di integrazione nelle prime fasi del ciclo di sviluppo.
- Implementazione e gestione del parco veicoli: diagnostica gli errori più comuni nella creazione del parco veicoli, i problemi di attivazione dei processi server, i problemi di scalabilità e gli errori di distribuzione che possono impedire il corretto funzionamento dei server di gioco.
- Diagnostica e monitoraggio del runtime: accedi alle istanze live della flotta per la risoluzione dei problemi in tempo reale, l'analisi delle prestazioni e la convalida della configurazione per risolvere i problemi che influiscono sull'esperienza dei giocatori.

Usa queste risorse insieme alle funzionalità GameLift di monitoraggio e registrazione di Amazon per mantenere prestazioni e disponibilità ottimali per i tuoi giochi multiplayer.

Configurazione per lo sviluppo iterativo con Anywhere Amazon GameLift Servers

Amazon GameLift Servers fornisce strumenti e soluzioni per aiutarti a configurare un ambiente di test ospitato da utilizzare durante lo sviluppo del gioco. Con questi strumenti, puoi creare un ambiente che rispecchi l'esperienza dei giocatori nel mondo reale dell'hosting gestito Amazon GameLift Servers e supporti un processo di sviluppo rapido e iterativo.

Con un ambiente di test separato, rimuovi il sovraccarico di una Amazon GameLift Servers flotta gestita durante i test. Non è più necessario caricare ogni nuova iterazione di build del server di gioco, creare una nuova flotta e attendere più di 15 minuti prima che si attivi. Puoi invece creare una nuova build, aggiornare rapidamente la flotta di test con la nuova build, avviarla e iniziare i test.

Utilizzando una flotta Amazon GameLift Servers Anywhere, puoi configurare un ambiente di test utilizzando un dispositivo locale, come la tua workstation di sviluppo. Puoi anche configurare un ambiente di test utilizzando una risorsa di hosting basata su cloud.

Configura un ambiente di test Anywhere per sviluppare e testare una serie di scenari, tra cui questi:

- Verifica l'integrazione del tuo server di gioco con l'SDK Amazon GameLift Servers del server. Puoi eseguire il test anche senza un client di gioco funzionante utilizzando le chiamate AWS CLI per avviare nuove sessioni di gioco e tenere traccia degli eventi delle sessioni di gioco.
- Metti alla prova le interazioni tra il client di gioco, il servizio di backend e il Amazon GameLift Servers servizio mentre sviluppi i componenti per il gioco. Perfeziona l'esperienza del giocatore per partecipare a una partita.
- Sperimenta con il design del tuo FlexMatch matchmaker. Prova le varianti del set di regole e altre implementazioni delle funzionalità di matchmaking. Configura e testa il matchmaking backfill.
- Prova altre funzionalità di Amazon GameLift Servers hosting, come le impostazioni di configurazione del runtime (con l'Amazon GameLift ServersAgent) per la gestione del ciclo di vita dei server di gioco.
- Crea, testa e ripeti rapidamente per convalidare tutti gli aspetti dell'esperienza di gioco, comprese le interazioni multigiocatore, in un ambiente live ospitato.

Successivamente, mentre prepari il gioco per il lancio, ti consigliamo di aggiungere flotte Amazon GameLift Servers gestite per ottimizzare le configurazioni di hosting e testare scenari aggiuntivi, tra cui i seguenti:

- Sperimenta e testa i design delle code delle sessioni di gioco, incluso l'uso di flotte con più sedi, flotte Spot e On-Demand e più tipi di istanze.
- Prova le opzioni di posizionamento delle sessioni di gioco con flotte gestite, incluso l'uso di politiche di latenza opzionali e impostazioni di prioritizzazione della flotta.
- Configura la scalabilità della capacità per soddisfare la domanda dei giocatori, utilizzando opzioni di ridimensionamento automatico o manuale.
- Configura flotte AWS CloudFormation Amazon GameLift Servers gestite per gestire le risorse di hosting a lungo termine.

Fast Build Update Tool (solo per lo sviluppo)

Con EC2 le flotte gestite, per implementare un aggiornamento della build di un server di gioco, devi caricare ogni nuova build Amazon GameLift Servers e creare una nuova flotta per essa.

Il Fast Build Update Tool ti consente di aggirare questi passaggi durante lo sviluppo, risparmiando tempo e velocizzando l'iterazione dello sviluppo. Con questo strumento, puoi aggiornare rapidamente i file di build del gioco su tutti i computer di una flotta esistente. Lo strumento offre diverse opzioni: puoi sostituire un'intera build di gioco o modificare 6 file specifici e puoi gestire il riavvio dei processi del server di gioco dopo gli aggiornamenti. Puoi anche usarlo per aggiornare i singoli computer di una flotta.

Per scaricare il Fast Build Update Tool e saperne di più su come usarlo, visita il repository Amazon GameLift Servers Toolkit per [The Fast Build Update Tool](#) in Github.

Argomenti

- [Crea un ambiente di test basato sul cloud](#)
- [Configura i test locali con Amazon GameLift Servers Anywhere](#)
- [Testa la tua integrazione usando Amazon GameLift Servers Local](#)

Crea un ambiente di test basato sul cloud

Note

Questo argomento tratta i test iterativi per i giochi integrati con l'SDK del server per Amazon GameLift Servers la versione 5.x. Se il gioco utilizza la versione 4.x o precedente dell'SDK per server, consulta. [Testa la tua integrazione usando Amazon GameLift Servers Local](#)

Usa una flotta Amazon GameLift Servers Anywhere per creare e testare in modo iterativo i componenti del gioco in un ambiente ospitato basato su cloud. Crea una flotta Anywhere con risorse di hosting e una connessione al Amazon GameLift Servers servizio, esegui i server di gioco su di essa e testa le funzionalità di gioco secondo necessità.

Implementa una flotta Anywhere con l'agente Amazon GameLift Servers

Se la build del tuo server di gioco è integrata con Amazon GameLift Servers SDK 5.x o versione successiva, puoi distribuirla su una flotta Anywhere basata su cloud con l'Agent. Amazon GameLift Servers L'Agent è un processo in background che gestisce i cicli di vita dei server di gioco e altre attività su ogni computer di una flotta. Queste attività includono la registrazione del computer con una flotta Anywhere, l'acquisizione di un token di autenticazione e i processi del server di starting/

stopping gioco basati su una serie di istruzioni. L'agente è controllato dalla configurazione di runtime di una flotta, che puoi aggiornare in qualsiasi momento durante la vita del parco veicoli. (L'agente viene distribuito automaticamente nelle flotte EC2 gestite). [Per ulteriori informazioni e per scaricare l'agente, consulta il repository. Amazon GameLift Servers GitHub](#)

Configura test iterativi con Amazon EC2

Utilizza il flusso di lavoro guidato di questa [soluzione di Amazon GameLift Servers toolkit](#) per configurare un ambiente di hosting basato sul cloud che rispecchi l'esperienza di hosting gestito con Amazon GameLift Servers

Il GitHub repository fornisce un set di script che automatizzano la maggior parte dei processi per la configurazione di un ambiente di test con Anywhere and the Agent. Amazon GameLift Servers Fornisce inoltre indicazioni per aggiornare l'ambiente ogni volta che devi testare una nuova build di server di gioco. Puoi eseguire un singolo script che distribuisca un ambiente di test con una build di esempio del server di gioco, oppure puoi seguire ogni passaggio per configurarlo con la build del tuo server di gioco.

In questo flusso di lavoro, lavorerai interamente in Console di gestione AWS, utilizzando AWS CloudShell per eseguire script e completare attività da riga di comando.

Note

Per le attività di questo tutorial, è necessario un utente AWS account con autorizzazioni per i seguenti servizi: Amazon GameLift Servers, AWS CloudShell, Amazon S3 AWS Systems Manager, Amazon EC2 e. AWS Identity and Access Management Gli utenti con accesso a livello di amministratore all' AWS account dispongono già delle autorizzazioni richieste.

Il flusso di lavoro copre le seguenti attività:

- Package per un server di gioco creato per Amazon GameLift Servers. Il flusso di lavoro fornisce uno script per creare un server di gioco C++ di esempio, che è già stato integrato con l'SDK del server per la Amazon GameLift Servers versione 5.x ed è pronto per l'hosting. In alternativa, puoi lavorare con il tuo progetto di gioco se hai completato l'integrazione.
- Configura un bucket Amazon Simple Storage Service per archiviare build e dipendenze dei server di gioco. Man mano che produci nuove versioni delle tue build di gioco, puoi archivarle in S3 e utilizzare gli script per aggiornare la flotta di Anywhere per i test di gioco.

- Ottieni e costruisci l'agente. Amazon GameLift Servers L'agente gestisce i processi del server di gioco su una risorsa di hosting in base alla configurazione dell'utente. Utilizza la stessa logica e si comporta in modo identico all'hosting EC2 Amazon GameLift Servers gestito.
- Configura una flotta Anywhere per le tue risorse di hosting. Con una flotta Anywhere puoi utilizzare il Amazon GameLift Servers servizio per ospitare risorse che non sono gestite da Amazon GameLift Servers. In questo passaggio, configurerai anche la configurazione di runtime, che indica ad Amazon GameLift Servers Agent quando e come avviare i processi del server di gioco.
- Configura un'istanza Amazon EC2. Questo è il tuo ambiente di test per test iterativi. È molto più veloce utilizzare un'istanza EC2 standard anziché un'istanza completamente gestita Amazon GameLift Servers (ottimizzata per l'utilizzo a livello di produzione). Con un'istanza EC2 standard, puoi aggiornare rapidamente e continuamente il server di gioco secondo necessità.
- Implementa la build del server di gioco e Amazon GameLift Servers l'agente sull'istanza Amazon EC2. Il flusso di lavoro fornisce uno script che ottiene la versione più recente della build del gioco e tutte le dipendenze e la installa sulla tua istanza EC2. In questo flusso di lavoro, le dipendenze includono l'Amazon GameLift Servers agente e l'agente. CloudWatch
- Avvia l'Amazon GameLift Servers agente. Una volta installato, l'agente si avvia e inizia automaticamente a eseguire le istruzioni. Ciò include:
 - Registra l'istanza EC2 come elaborazione nella flotta Amazon GameLift Servers Anywhere.
 - Stabilisci una WebSocket connessione con il Amazon GameLift Servers servizio e ottieni la configurazione di runtime più recente.
 - Avvia i processi del server di gioco in base alle istruzioni nella configurazione di runtime. In questo flusso di lavoro, all'agente viene richiesto di avviare un singolo processo dell'eseguibile del server di gioco.
- Metti alla prova i tuoi scenari di gioco. Con l'ambiente di test configurato e l'ultima build del server di gioco installata, puoi iniziare i test. Il flusso di lavoro illustra diversi passaggi per il test, incluso l'avvio di una sessione di gioco. Accedi ai log del server di CloudWatch gioco per tenere traccia dei progressi man mano che la sessione di gioco si avvia e si prepara ad accettare giocatori.

Man mano che sviluppi i componenti del gioco, tra cui un client di gioco e un servizio di backend lato client, puoi includerli nei tuoi scenari di test. Usa un client di gioco per richiedere una sessione di gioco, recuperare le informazioni di connessione dal Amazon GameLift Servers servizio e quindi connetterti direttamente alla sessione di gioco.

- Implementa una nuova build del server di gioco e ripeti i test. Durante lo sviluppo del gioco, puoi generare nuove build di server di gioco, quindi distribuirle rapidamente nell'ambiente di test EC2

per i test. Caricali nel bucket Amazon S3 e poi usa gli script del flusso di lavoro per aggiornare l'ambiente di test.

Passa il tuo gioco a flotte gestite Amazon GameLift Servers

Dopo aver completato i test di sviluppo e pronto a prepararti per il lancio, questo è un buon momento per passare alle flotte Amazon GameLift Servers gestite. Usa le flotte gestite per perfezionare e testare le tue risorse di hosting di giochi. Implementa la tua soluzione di posizionamento delle sessioni di gioco (code e matchmaker), seleziona l'hardware di hosting (comprese le flotte Spot) e le sedi ottimali e scegli una strategia per aumentare la capacità. Potresti anche iniziare a AWS CloudFormation utilizzarla per gestire in modo più efficiente i cicli di vita di tutte le tue risorse di hosting di giochi, tra cui flotte, code e matchmaker.

La transizione da una flotta di test Anywhere basata sul cloud a una flotta gestita richiede uno sforzo minimo. Amazon GameLift Servers Non è necessario modificare alcun codice di gioco e puoi riutilizzare le stesse code e gli stessi matchmaker. Esegui le seguenti attività:

- Crea una risorsa di Amazon GameLift Servers compilazione. Con una flotta di test Anywhere, devi distribuire manualmente la build e le dipendenze del server di gioco su ogni computer della flotta. Con una flotta gestita, carica il pacchetto di build del gioco su Amazon GameLift Servers, che lo distribuirà automaticamente su tutti i computer della flotta. [Crea un server di gioco costruito per Amazon GameLift Servers](#) Per ulteriori informazioni su come impacchettare i file di build del gioco e creare una risorsa di compilazione con file in un bucket Amazon S3, consulta la pagina.
- Crea una flotta gestita. Crea una flotta utilizzando la console o la AWS CLI, specificando una flotta gestita da EC2. Questo tipo di parco macchine richiede impostazioni di configurazione aggiuntive, tra cui la specificazione delle risorse di compilazione e dei tipi di istanze. Puoi utilizzare la stessa configurazione di runtime per gestire il ciclo di vita dei server di gioco su ogni computer della flotta. Vedi [Crea una EC2 flotta Amazon GameLift Servers gestita](#) i dettagli sulla creazione di una flotta gestita.
- Reindirizza gli alias del parco veicoli (opzionale). Se configuri alias da utilizzare con le tue flotte Anywhere, puoi riutilizzare gli stessi alias per le flotte gestite. Vedi [Crea un Amazon GameLift Servers alias](#) per i dettagli sulla creazione o l'aggiornamento di un alias.

Configura i test locali con Amazon GameLift Servers Anywhere

Note

Questo argomento tratta dei test locali per i giochi integrati con l'SDK del server per la Amazon GameLift Servers versione 5.x. Se il gioco utilizza la versione 4.x o precedente dell'SDK per server, consulta. [Testa la tua integrazione usando Amazon GameLift Servers Local](#)

Usa una flotta Amazon GameLift Servers Anywhere e il tuo hardware per creare e testare in modo iterativo i componenti del gioco in un ambiente ospitato simulato. Configura una flotta Anywhere e registra un dispositivo locale per stabilire una connessione al Amazon GameLift Servers servizio. Installa il server di gioco integrato sul dispositivo, avvia un processo del server di gioco e verifica le funzionalità di gioco secondo necessità. Puoi aggiornare la build del server di gioco tutte le volte che vuoi per testare ogni nuova build.

Con una flotta Anywhere, puoi eseguire i test utilizzando la AWS CLI o con script di test. Se hai integrato un client di gioco con Amazon GameLift Servers, puoi eseguire il client sullo stesso dispositivo locale o su un dispositivo diverso.

I test locali con una flotta Anywhere sono particolarmente utili per testare l'integrazione del server di gioco con Amazon GameLift Servers. Hai piena visibilità su tutte le attività di hosting sul computer locale, nonché sugli eventi e sui dati di registrazione.

Note

Stai usando il Amazon GameLift Servers plugin per Unreal Engine o Unity? Questi strumenti includono flussi di lavoro guidati per l'impostazione di test locali con una flotta Anywhere. Segui la documentazione per [Plugin per Unity: configura i test locali con Anywhere Amazon GameLift Servers](#) o [Plugin per Unreal: ospita il tuo gioco localmente con Anywhere Amazon GameLift Servers](#).

Argomenti

- [Configura una flotta Anywhere locale](#)
- [Aggiorna e installa il tuo server di gioco](#)
- [Verifica l'attività della sessione di gioco](#)

- [Esegui un'iterazione sul tuo server di gioco](#)
- [Trasforma il tuo gioco in flotte Amazon GameLift Servers gestite](#)

Configura una flotta Anywhere locale

Segui questi passaggi per creare una flotta Anywhere per la tua workstation locale. Per istruzioni dettagliate sull'utilizzo della AWS CLI o del modulo Amazon GameLift Servers, Console di gestione AWS consulta. [Crea un Amazon GameLift Servers Flotta ovunque](#)

Per creare la flotta Anywhere

1. Crea una posizione personalizzata per la tua workstation locale. (AWS CLI o console). Una posizione personalizzata è semplicemente un'etichetta per la risorsa di elaborazione che intendi includere nella tua flotta Anywhere. I nomi delle località personalizzate devono iniziare con custom-. Ad esempio: custom-my_laptop. Per informazioni, consulta [Creare una posizione personalizzata](#).
2. Crea una flotta Anywhere (AWS CLI o console). In questo passaggio, crea la risorsa del parco veicoli con la posizione personalizzata per la workstation locale. Per informazioni, consulta [Crea una flotta Anywhere](#).

Prendi nota dell'ID o del valore ARN del nuovo parco veicoli. Tale valore servirà nella fase successiva.

3. Registra la tua workstation locale come unità di calcolo della flotta (solo AWS CLI). Una flotta Anywhere deve disporre di almeno una risorsa di elaborazione per ospitare i server di gioco. Per informazioni, consulta [Aggiungi un computer alla flotta](#). Per aggiungere un computer alla flotta, sono necessarie le seguenti informazioni:
 - Un nome di calcolo. Ogni calcolo di un parco dati deve avere un nome univoco.
 - L'identificatore della flotta Anywhere. Puoi usare FleetID o FleetArn.
 - Le informazioni sulla connessione del computer. Specifica un IpAddress oppure il valore DnsName. Ecco come i client Amazon GameLift Servers di gioco si connetteranno ai server di gioco.
 - Una posizione personalizzata nella flotta Anywhere.

Prendi nota del valore `GameLiftServiceSdkEndpoint` restituito. Questo valore ti servirà quando aggiorni il tuo server di gioco per farlo funzionare su una flotta Anywhere.

Aggiorna e installa il tuo server di gioco

Questa attività presuppone che tu abbia già integrato una build di server di gioco con il Amazon GameLift Servers server SDK 5.x. Il processo di integrazione prevede l'aggiunta di codice al server di gioco in modo che possa interagire con il Amazon GameLift Servers servizio per avviare e gestire le sessioni di gioco.

Per una flotta Anywhere, devi configurare manualmente alcune impostazioni del server di gioco. In una flotta Amazon GameLift Servers gestita, queste impostazioni vengono configurate automaticamente.

Per preparare il tuo server di gioco per una flotta Anywhere

1. Ottieni un token di autenticazione. Il tuo server di gioco deve includere un token di autenticazione in ogni comunicazione con il Amazon GameLift Servers servizio. Amazon GameLift Servers token di autenticazione sono di breve durata e devono essere aggiornati regolarmente.

È consigliabile creare uno script per completare le seguenti attività:

- Richiama l'azione AWS CLI. `get-compute-auth-token`
- Memorizza il valore del token restituito dove i processi del server di gioco possono recuperarlo, ad esempio in una variabile di ambiente sul calcolo locale.

Installa lo script con il tuo server di gioco sul computer. Imposta lo script in modo che venga eseguito prima di iniziare il primo processo del server di gioco. Mentre i processi del server di gioco sono attivi, esegui lo script regolarmente per mantenere un token di autenticazione valido. Tutti i processi del server di gioco sul computer possono utilizzare lo stesso token di autenticazione.

2. Aggiorna il codice del server Amazon GameLift Servers di gioco. Quando hai integrato il codice del server di gioco con l'SDK del server per Amazon GameLift Servers, hai aggiunto un invito all'azione `InitSdk()`. Quando il server di gioco funziona su una flotta Anywhere, questa chiamata richiede parametri server aggiuntivi. Per ulteriori informazioni, consulta [Inizializza il processo del server](#) la sezione [Server SDK 5.x per Amazon GameLift Servers](#) relativa al linguaggio di sviluppo in uso. I parametri del server sono:

- `WebSocketUrl`— Imposta questo parametro sul `GameLiftServiceSdkEndpoint` valore che viene restituito quando registri un calcolo nella flotta.

- `hostId`— Imposta questo parametro sul nome di calcolo che specifichi quando registri un calcolo con la flotta Anywhere.
- `fleetId`— Imposta questo parametro sull'ID della flotta Anywhere.
- `authToken`— Imposta questo parametro sul token che viene restituito in risposta a una richiesta di recupero di un token di autenticazione per un calcolo.
- `processId`— Imposta questo parametro per identificare un processo del server di gioco in esecuzione sul computer locale. Ogni processo simultaneo del server di gioco deve avere un ID di processo univoco.

I valori dei parametri del server utilizzati da ogni processo del server di gioco devono essere specifici per l'elaborazione della flotta Anywhere su cui è in esecuzione il processo. Per i dettagli su come ottenere i valori appropriati per un calcolo, consulta [Aggiungi un computer alla flotta](#). Come procedura consigliata `websocketUrl` `hostId` `fleetId`, imposta `authToken` come variabili di ambiente nell'elaborazione locale. Tutti i processi del server eseguiti sul computer utilizzeranno questi valori.

3. Installa la build del server di gioco sul computer locale. Includi tutte le dipendenze necessarie per far funzionare il server di gioco.
4. Avvia uno o più processi del server di gioco in esecuzione sul computer locale. Quando il processo del server di gioco richiama l'azione SDK del `serverProcessReady()`, il processo è pronto per ospitare una sessione di gioco.

Verifica l'attività della sessione di gioco

Verifica l'integrazione del tuo server di gioco lavorando con le sessioni di gioco. Se non disponi di un client di gioco integrato con Amazon GameLift Servers funzionalità, puoi utilizzare la AWS CLI per avviare le sessioni di gioco. Prova i seguenti scenari:

- Creare una sessione di gioco. [create-game-session](#) Comando di chiamata (o operazione [CreateGameSession](#) API). Specificate l'ID e la posizione personalizzata della vostra flotta Anywhere. Questa chiamata restituisce un identificatore univoco per la nuova sessione di gioco.
- Controlla lo stato della sessione di gioco. [describe-game-sessions](#) Comando di chiamata (o azione [DescribeGameSessions](#) API). Specificate l'ID della sessione di gioco. Questa chiamata restituisce informazioni dettagliate sulla sessione di gioco, incluso lo stato della sessione di gioco. Le sessioni di gioco in stato Attivo sono pronte per consentire ai giocatori di connettersi. Per ottenere un

elenco di tutte le sessioni di gioco per la flotta, chiama il [list-game-sessions](#) comando (o l'azione [ListGameSessions](#) API).

- Connect alla sessione di gioco. Se il tuo client di gioco ha la possibilità di partecipare a una sessione di gioco, utilizza le informazioni di connessione incluse nelle informazioni sulla sessione di gioco.

Esegui un'iterazione sul tuo server di gioco

Puoi usare la stessa Anywhere flotta e lo stesso calcolo per testare altre versioni della build del tuo server di gioco.

1. Pulisci il tuo esistente **GameSession**. Se il processo del server di gioco si blocca o non chiama `ProcessEnding()`, Amazon GameLift Servers pulisce `GameSession` dopo che il server di gioco ha interrotto l'invio dei controlli di integrità.
2. Genera una nuova build del server di gioco. Apporta modifiche al tuo server di gioco e crea una build rivista.
3. Aggiorna la build del server di gioco sul tuo computer locale. La tua flotta Anywhere precedente è ancora attiva e il tuo laptop è ancora registrato come risorsa di elaborazione nel parco.
4. Ottieni un token di autorizzazione aggiornato. Chiama il comando [get-compute-auth-token](#) CLI e archivia il token sul calcolo locale.
5. Avvia uno o più processi del server di gioco in esecuzione sul computer locale. Quando il processo del server di gioco chiama `ProcessReady()`, è pronto per essere utilizzato per i test.

Trasforma il tuo gioco in flotte Amazon GameLift Servers gestite

Dopo aver completato i test di sviluppo e pronto a prepararti per il lancio, questo è un buon momento per passare alle flotte Amazon GameLift Servers gestite. Usa le flotte gestite per perfezionare e testare le tue risorse di hosting di giochi. Implementa la tua soluzione di posizionamento delle sessioni di gioco (code e matchmaker), seleziona l'hardware di hosting (comprese le flotte Spot) e le sedi ottimali e scegli una strategia per aumentare la capacità. Potresti anche iniziare a AWS CloudFormation utilizzarla per gestire in modo più efficiente i cicli di vita di tutte le tue risorse di hosting di giochi, tra cui flotte, code e matchmaker.

È necessario apportare alcune modifiche minori per passare da una flotta di test Anywhere locale a una flotta gestita. Amazon GameLift Servers Puoi riutilizzare le stesse code e gli stessi matchmaker. Esegui le seguenti attività:

- Cambia la chiamata in codice del server di gioco in **InitSdk()**. Rimuovi i parametri del server. Per una flotta gestita, tiene traccia Amazon GameLift Servers automaticamente di queste informazioni.
- Crea una risorsa di Amazon GameLift Servers compilazione. Con una flotta di test Anywhere, devi distribuire manualmente la build e le dipendenze del server di gioco su ogni computer della flotta. Con una flotta gestita, puoi creare e caricare il tuo pacchetto di build di gioco su Amazon GameLift Servers, che lo distribuisce automaticamente su tutti i computer della flotta. [Crea un server di gioco costruito per Amazon GameLift Servers](#) Per ulteriori informazioni su come impacchettare i file di build del gioco e creare una risorsa di compilazione con file in un bucket Amazon S3, consulta la pagina. Non includere script che registrano un'elaborazione e ottengono un token di autenticazione, poiché gestisce Amazon GameLift Servers automaticamente queste attività con flotte gestite.
- Crea una flotta gestita. Crea una flotta utilizzando la console o la AWS CLI, specificando una flotta gestita da EC2. Questo tipo di parco macchine richiede impostazioni di configurazione aggiuntive, tra cui la specificazione delle risorse di compilazione e dei tipi di istanze. Dovete tutti impostare una configurazione di runtime per gestire il ciclo di vita dei server di gioco su ogni computer della flotta. [Crea una EC2 flotta Amazon GameLift Servers gestita](#) Per ulteriori informazioni sulla creazione di una flotta gestita, consulta la sezione.
- Reindirizza gli alias del parco veicoli (opzionale). Se configuri alias da utilizzare con le tue flotte Anywhere, puoi riutilizzare gli stessi alias per le flotte gestite. Vedi [Crea un Amazon GameLift Servers alias](#) per i dettagli sulla creazione o l'aggiornamento di un alias.

Testa la tua integrazione usando Amazon GameLift Servers Local

Note

Questo argomento tratta i test per i giochi integrati con l'SDK del server solo per le Amazon GameLift Servers versioni 3.x e 4.x. Il pacchetto SDK del server include una versione compatibile di Local. Amazon GameLift Servers Se utilizzi la versione 5.x dell'SDK del server, consulta [Test locali con GameLift Server SDK 5.x per i test](#) locali con una flotta Anywhere. Amazon GameLift Servers [Per indicazioni sulla migrazione da SDK 4.x a 5.x, consulta Migrate to Server SDK 5.x. GameLift](#)

Utilizza Amazon GameLift Servers Local per eseguire una versione limitata del servizio Amazon GameLift Servers gestito su un dispositivo locale e testare l'integrazione dei giochi. Questo strumento è utile quando si esegue lo sviluppo iterativo sull'integrazione dei giochi. L'alternativa, vale a dire

caricare ogni nuova build Amazon GameLift Servers e configurare una flotta per ospitare il gioco, può richiedere diverse o più volte ogni volta.

Con Amazon GameLift Servers Local, è possibile verificare quanto segue:

- Il server di gioco è correttamente integrato con l'SDK del server e comunica correttamente con il servizio Amazon GameLift Servers per avviare nuove sessioni di gioco, accettare nuovi giocatori e fornire rapporti sull'integrità e sullo stato.
- Il client di gioco è correttamente integrato con l' AWS SDK per Amazon GameLift Servers ed è in grado di recuperare informazioni sulle sessioni di gioco esistenti, avviare nuove sessioni di gioco, unire giocatori alle partite e connettersi alla sessione di gioco.

Amazon GameLift ServersLocal è uno strumento da riga di comando che avvia una versione autonoma del servizio gestito. Amazon GameLift Servers Local fornisce anche un registro degli eventi in esecuzione relativo all'inizializzazione dei processi del server, ai controlli di integrità e alle chiamate e risposte delle API. Amazon GameLift Servers Local riconosce un sottoinsieme delle azioni AWS SDK per. Amazon GameLift Servers Puoi effettuare chiamate dal AWS CLI o dal tuo client di gioco. Tutte le operazioni API vengono eseguite a livello locale proprio come avviene nel servizio Web Amazon GameLift Servers.

Ogni processo del server dovrebbe ospitare solo una singola sessione di gioco. La sessione di gioco è l'eseguibile che usi per connetterti a Amazon GameLift Servers Local. Quando la sessione di gioco è completata, dovresti chiamare `GameLiftServerSDK::ProcessEnding` e poi uscire dal processo. Quando esegui il test localmente con Amazon GameLift Servers Local, puoi avviare più processi sul server. Ogni processo si conatterà a Amazon GameLift Servers Local. Puoi quindi creare una sessione di gioco per ogni processo del server. Al termine della sessione di gioco, il processo del server di gioco dovrebbe chiudersi. È quindi necessario avviare manualmente un altro processo del server.

Amazon GameLift Serverslocal supporta quanto segue APIs:

- `CreateGameSession`
- `CreatePlayerSession`
- `CreatePlayerSessions`
- `DescribeGameSessions`
- `DescribePlayerSessions`

Configurazione Amazon GameLift Servers locale

Amazon GameLift Servers Local viene fornito come un file `.jar` eseguibile integrato con l'[SDK del server](#). Può essere eseguito su Windows o Linux e utilizzato con qualsiasi linguaggio supportato da Amazon GameLift Servers. Per la versione SDK 4.0.2, puoi scaricarla dalle versioni [ufficiali GitHub](#). Il `GameLiftLocal file.jar` è incluso nell'elemento `-4.0.2.zip`. `GameLift-CSharp-ServerSDK`

Prima di eseguire Local, è necessaria l'installazione di quanto segue.

- Una build dell'SDK del server per le versioni da 3.1.5 a 4.x. Amazon GameLift Servers
- Java 8

Prova un server di gioco

Se desideri testare solo il tuo server di gioco, puoi utilizzarlo AWS CLI per simulare le chiamate dei client di gioco al servizio Amazon GameLift Servers locale. In questo modo si verifica che il server di gioco funziona come previsto nelle condizioni seguenti:

- Il server di gioco si avvia correttamente e inizializza l'SDK del server per. Amazon GameLift Servers
- Nell'ambito del processo di avvio, il server di gioco comunica ad Amazon GameLift Servers che il server è pronto per l'hosting di sessioni di gioco.
- Il server di gioco invia lo stato di integrità a Amazon GameLift Servers ogni minuto durante l'esecuzione.
- Il server di gioco risponde alle richieste di avvio di una nuova sessione di gioco.

1. Avviare Amazon GameLift Servers Local.

Aprire una finestra del prompt dei comandi, accedere alla directory contenente il file `GameLiftLocal.jar` ed eseguire il file. Per impostazione predefinita, Local riceve le richieste provenienti dai client di gioco sulla porta 8080. Per specificare un numero di porta diverso, utilizzare il parametro `-p`, come indicato nell'esempio seguente:

```
java -jar GameLiftLocal.jar -p 9080
```

Una volta avviato Local, è possibile visualizzare i log che indicano che sono stati avviati due server locali, di cui uno rileva il server di gioco e l'altro il client di gioco o la AWS CLI. I log

continuano a fornire rapporti sull'attività dei due server locali, comprese le comunicazioni da e verso i componenti di gioco.

2. Avviare il server di gioco.

Avviare il server di gioco integrato con Amazon GameLift Servers a livello locale. Non è necessario modificare l'endpoint per il server di gioco.

Nella finestra del prompt dei comandi di Local, i messaggi di log indicano che il server di gioco è connesso al servizio Amazon GameLift Servers Local. Ciò significa che il server di gioco ha inizializzato con successo l'SDK del server per (con). Amazon GameLift Servers `InitSDK()` Ha chiamato `ProcessReady()` con i percorsi di log mostrati e, in caso di esito positivo, è pronto per l'hosting di una sessione di gioco. Mentre il server di gioco è in esecuzione, Amazon GameLift Servers registra ogni rapporto sullo stato di integrità del server di gioco. Il seguente esempio di messaggistica di log mostra un server di gioco integrato:

```
16:50:53,217 INFO || - [SDKListenerImpl] nioEventLoopGroup-3-1 - SDK
connected: /127.0.0.1:64247
16:50:53,217 INFO || - [SDKListenerImpl] nioEventLoopGroup-3-1 - SDK pid is 17040,
sdkVersion is 3.1.5 and sdkLanguage is CSharp
16:50:53,217 INFO || - [SDKListenerImpl] nioEventLoopGroup-3-1 - NOTE: Only SDK
versions 3.1.5 and above are supported in GameLiftLocal!
16:50:53,451 INFO || - [SDKListenerImpl] nioEventLoopGroup-3-1 - onProcessReady
received from: /127.0.0.1:64247 and ackRequest requested? true
16:50:53,543 INFO || - [SDKListenerImpl] nioEventLoopGroup-3-1 - onProcessReady
data: logPathsToUpload: "C:\\game\\logs"
logPathsToUpload: "C:\\game\\error"
port: 1935

16:50:53,544 INFO || - [HostProcessManager] nioEventLoopGroup-3-1 - Registered new
process true, true,
16:50:53,558 INFO || - [SDKListenerImpl] nioEventLoopGroup-3-1 - onReportHealth
received from /127.0.0.1:64247 with health status: healthy
```

Potenziati messaggi di errore e avviso sono i seguenti:

- Errore: "ProcessReady non ho trovato un processo con PID:<process ID>! È stato invocato `initSDK()`?"
- Avviso: «Lo stato del processo esiste già per il processo con PID:<process ID>! `ProcessReady(...)` è stato invocato più di una volta?»

3. Avvia il. AWS CLI

Una volta che il server di gioco ha completato la chiamata a `ProcessReady()`, è possibile iniziare a effettuare chiamate client. Aprire un'altra finestra del prompt dei comandi e avviare lo strumento AWS CLI. Per impostazione AWS CLI predefinita, utilizza l'endpoint del servizio Amazon GameLift Servers Web. È necessario sostituirlo con l'endpoint di Local in ogni richiesta con il parametro `--endpoint-url`, come illustrato nella seguente richiesta di esempio.

```
AWS gamelift describe-game-sessions --endpoint-url http://localhost:9080 --fleet-id fleet-123
```

[Nella finestra del AWS CLI prompt dei AWS gamelift comandi, i comandi generano risposte come documentato nel Command Reference.AWS CLI](#)

4. Creare una sessione di gioco.

Con AWS CLI, invia una richiesta [CreateGameSession\(\)](#). La richiesta deve seguire la sintassi prevista. Per Local, il parametro `FleetId` può essere impostato su qualsiasi stringa valida (`^fleet-\S+`).

```
AWS gamelift create-game-session --endpoint-url http://localhost:9080 --maximum-player-session-count 2 --fleet-id fleet-1a2b3c4d-5e6f-7a8b-9c0d-1e2f3a4b5c6d
```

Nella finestra del prompt dei comandi di Local, i messaggi di log indicano che Amazon GameLift Servers Local ha inviato al server di gioco un callback `onStartGameSession`. Se una sessione di gioco è creata, il server di gioco risponde richiamando `ActivateGameSession`.

```
13:57:36,129 INFO || - [SDKInvokerImpl]
    Thread-2 - Finished sending event to game server to start a game session:
    arn:aws:gamelift:local::gamesession/
    fleet-1a2b3c4d-5e6f-7a8b-9c0d-1e2f3a4b5c6d/gsess-ab423a4b-b827-4765-
    aea2-54b3fa0818b6.
    Waiting for ack response.13:57:36,143 INFO || - [SDKInvokerImpl]
    Thread-2 - Received ack response: true13:57:36,144 INFO || -
    [CreateGameSessionDispatcher] Thread-2 - GameSession with id:
    arn:aws:gamelift:local::gamesession/
    fleet-1a2b3c4d-5e6f-7a8b-9c0d-1e2f3a4b5c6d/gsess-ab423a4b-b827-4765-
    aea2-54b3fa0818b6
    created13:57:36,227 INFO || - [SDKListenerImpl]
    nioEventLoopGroup-3-1 - onGameSessionActivate received
    from: /127.0.0.1:60020 and ackRequest
```

```
requested? true13:57:36,230 INFO || - [SDKListenerImpl]
nioEventLoopGroup-3-1 - onGameSessionActivate data: gameId:
"arn:aws:gamelift:local::gamesession/
fleet-1a2b3c4d-5e6f-7a8b-9c0d-1e2f3a4b5c6d/gsess-abcdef12-3456-7890-abcd-
ef1234567890"
```

Nella AWS CLI finestra, Amazon GameLift Servers risponde con un oggetto della sessione di gioco che include un ID della sessione di gioco. Si noti che lo stato della nuova sessione di gioco è **Activating** (In fase di attivazione). Lo stato diventa **Attivo** una volta **ActivateGameSession** richiamato il server di gioco. Se vuoi vedere lo stato modificato, usa il comando AWS CLI per chiamare **DescribeGameSessions()**

```
{
  "GameSession": {
    "Status": "ACTIVATING",
    "MaximumPlayerSessionCount": 2,
    "FleetId": "fleet-1a2b3c4d-5e6f-7a8b-9c0d-1e2f3a4b5c6d",
    "GameSessionId": "arn:aws:gamelift:local::gamesession/
fleet-1a2b3c4d-5e6f-7a8b-9c0d-1e2f3a4b5c6d/gsess-abcdef12-3456-7890-abcd-
ef1234567890",
    "IpAddress": "127.0.0.1",
    "Port": 1935
  }
}
```

Prova un server e un client di gioco

Per verificare l'integrazione completa dei giochi, compresa la connessione dei giocatori ai giochi, è possibile eseguire sia il server sia il client di gioco a livello locale. In questo modo è possibile testare le chiamate programmatiche dal client di gioco ad Amazon GameLift Servers Local. È possibile verificare le seguenti azioni:

- Il client di gioco sta effettuando correttamente le richieste AWS SDK al servizio Amazon GameLift Servers locale, tra cui la creazione di sessioni di gioco, il recupero di informazioni sulle sessioni di gioco esistenti e la creazione di sessioni di gioco.
- Il server di gioco convalida correttamente i giocatori quando tentano di partecipare a una sessione di gioco. Per i giocatori convalidati, il server di gioco potrebbe recuperare i dati giocatore (se la funzione è implementata).
- Il server di gioco segnala l'interruzione di una connessione quando un giocatore esce dal gioco.

- Il server di gioco segnala la fine di una sessione di gioco.

1. Avviare Amazon GameLift Servers Local.

Aprire una finestra del prompt dei comandi, accedere alla directory contenente il file *GameLiftLocal.jar* ed eseguire il file. Per impostazione predefinita, Local riceve le richieste provenienti dai client di gioco sulla porta 8080. Per specificare un numero di porta diverso, utilizzare il parametro `-p`, come indicato nell'esempio seguente.

```
./gamelift-local -p 9080
```

Una volta avviato Local, è possibile visualizzare i log che indicano che sono stati avviati due server locali, di cui uno rileva il server di gioco e l'altro il client di gioco o la AWS CLI.

2. Avviare il server di gioco.

Avviare il server di gioco integrato con Amazon GameLift Servers a livello locale. Per ulteriori informazioni sui log dei messaggi, consultare [Prova un server di gioco](#).

3. Configurare il client di gioco per Local e avviarlo.

Per utilizzare il client di gioco con il servizio Amazon GameLift Servers Local, occorre apportare le seguenti modifiche alla configurazione del client di gioco, come descritto nella sezione [Configura l'Amazon GameLift ServersAPI](#):

- Modificare l'oggetto `ClientConfiguration` in modo che punti all'endpoint Local, ad esempio `http://localhost:9080`.
- Impostare un valore di ID del parco istanze di destinazione. Per Local, non è necessario un vero e proprio ID del parco istanze; impostare il parco istanze di destinazione su qualsiasi stringa valida (`^fleet-\S+`), ad esempio `fleet-1a2b3c4d-5e6f-7a8b-9c0d-1e2f3a4b5c6d`.
- AWS Imposta le credenziali. Per Local, non occorrono vere e proprie credenziali AWS ; è possibile impostare la chiave di accesso e la chiave segreta su qualsiasi stringa.

Nella finestra del prompt dei comandi di Local, una volta avviato il client di gioco, i messaggi di log dovrebbero indicare che ha inizializzato `GameLiftClient` e ha comunicato con il servizio Amazon GameLift Servers.

4. Eseguire il test delle chiamate del client di gioco al servizio Amazon GameLift Servers.

Verificare che il client di gioco esegua una o tutte le seguenti chiamate API:

- [CreateGameSession\(\)](#)
- [DescribeGameSessions\(\)](#)
- [CreatePlayerSession\(\)](#)
- [CreatePlayerSessions\(\)](#)
- [DescribePlayerSessions\(\)](#)

Nella finestra del prompt dei comandi di Local, solo le chiamate a `CreateGameSession()` determinano messaggi di log. I messaggi di log mostrano quando Amazon GameLift Servers Local richiede al server di gioco di avviare una sessione di gioco (callback `onStartGameSession`) e ottiene una `ActivateGameSession` riuscita quando il server di gioco la richiama. Nella finestra della AWS CLI, tutte le chiamate API determinano risposte o messaggi di errore come documentato.

5. Verificare che il server di gioco convalidi le nuove connessioni dei giocatori.

Dopo aver creato una sessione di gioco e una sessione giocatore, stabilire una connessione diretta alla sessione di gioco.

Nella finestra del prompt dei comandi di Local, i messaggi di log devono mostrare che il server di gioco ha inviato una richiesta `AcceptPlayerSession()` per convalidare la nuova connessione dei giocatori. Se usi il comando AWS CLI `callDescribePlayerSessions()`, lo stato della sessione del giocatore dovrebbe cambiare da `Riservato` ad `Attivo`.

6. Verificare che il server di gioco riporti lo stato del gioco e dei giocatori al servizio Amazon GameLift Servers.

Perché Amazon GameLift Servers gestisca la richiesta dei giocatori e riporti correttamente i parametri, il server di gioco deve segnalare diversi stati ad Amazon GameLift Servers. Verificare che Local registri gli eventi correlati alle operazioni seguenti. Potresti anche voler utilizzare il per AWS CLI tenere traccia delle modifiche allo stato.

- Il giocatore si disconnette da una sessione di gioco: i messaggi di registro Amazon GameLift Servers locali dovrebbero mostrare che il server di gioco effettua chiamate `RemovePlayerSession()`. Le chiamate AWS CLI a `DescribePlayerSessions()` riflettono una modifica dello stato da `Active` a `Completed`.

È inoltre possibile chiamare `DescribeGameSessions()` per verificare che il numero di giocatori della sessione di gioco corrente diminuisce di uno.

- Fine della sessione di gioco: i messaggi di registro Amazon GameLift Servers locali dovrebbero mostrare che il server di gioco effettua chiamate `TerminateGameSession()`.

Note

La guida precedente era quella di chiamare al `TerminateGameSession()` termine di una sessione di gioco. Questo metodo è obsoleto con Amazon GameLift Servers Server SDK v4.0.1. Consultare [Termina una sessione di gioco](#).

- Il processo del server è terminato: i messaggi di registro Amazon GameLift Servers locali dovrebbero mostrare che il server di gioco effettua chiamate. `ProcessEnding()` Una AWS CLI chiamata a `DescribeGameSessions()` dovrebbe riflettere una modifica dello stato da `Active` a `Terminated` (o `Terminating`).

Varianti con locale

Quando si utilizza Amazon GameLift Servers Local, è necessario ricordare quanto segue:

- A differenza del servizio Web Amazon GameLift Servers, Local non monitora lo stato di integrità dei server e avvia il callback `onProcessTerminate`. Local interrompe semplicemente la registrazione dei rapporti sullo stato per il server di gioco.
- Per le chiamate all' AWS SDK, la flotta non IDs viene convalidata e può essere qualsiasi valore di stringa che soddisfi i requisiti del parametro `()^fleet-\S+`.
- Le sessioni di gioco IDs create con Local hanno una struttura diversa. Includono la stringa `local`, come mostrato qui:

```
arn:aws:gamelift:local::gamesession/fleet-123/gsess-56961f8e-  
db9c-4173-97e7-270b82f0daa6
```

Debug dei problemi del parco istanze di Amazon GameLift Servers

Questo argomento fornisce indicazioni su come risolvere i problemi relativi alle EC2 flotte Amazon GameLift Servers gestite.

Problemi di creazione del parco istanze

Quando crei una EC2 flotta gestita, il Amazon GameLift Servers servizio avvia un flusso di lavoro che crea la flotta, distribuisce EC2 istanze con la build del server di gioco installata e avvia i processi del server di gioco su ciascuna istanza. Per una descrizione dettagliata, consulta [Una flotta non può ospitare sessioni di gioco e giocatori finché non raggiunge lo stato Attivo](#). Una flotta non può ospitare sessioni di gioco e giocatori finché non raggiunge lo stato Attivo.

Puoi eseguire il debug dei problemi che impediscono alle flotte di diventare attive identificando la fase di creazione della flotta in cui si è verificato il problema ed esaminando gli eventi e i registri di creazione della flotta. Se i registri non offrono informazioni utili, è possibile che il problema sia dovuto a un errore interno del servizio. In questa situazione, prova a creare nuovamente la flotta. Se il problema persiste, prova a caricare nuovamente la build del gioco (per risolvere il possibile danneggiamento del file). Puoi anche contattare il supporto Amazon GameLift Servers o pubblicare una domanda nel forum.

Scaricamento e convalida della build

Durante questa fase, Amazon GameLift Servers ottiene la build del server di gioco caricata, estrae i file ed esegue gli eventuali script di installazione. Se la creazione della flotta fallisce durante queste fasi, consulta gli eventi e i registri della flotta per individuare il problema. Tra le cause possibili sono incluse:

- Amazon GameLift Servers non riesco a ottenere il file di build compresso (evento).
`FLEET_BINARY_DOWNLOAD_FAILED` Verifica che sia possibile accedere alla posizione di archiviazione della build, che tu stia creando una flotta nella Regione AWS stessa della build e che Amazon GameLift Servers disponga delle autorizzazioni corrette per accedervi.
- Amazon GameLift Servers non riesco a estrarre i file di build (evento `FLEET_CREATION_EXTRACTING_BUILD`).
- Uno script di installazione nei file di build non è stato completato correttamente (evento `FLEET_CREATION_FAILED_INSTALLER`).

Costruire risorse per la flotta

I problemi durante questa fase di solito riguardano l'allocazione e l'impiego delle risorse della flotta. Tra le cause possibili sono incluse:

- Il tipo di istanza richiesto non è disponibile.
- Il tipo di parco veicoli richiesto (Spot o On-Demand) non è disponibile.

Attivazione dei processi del server di gioco

Durante questa fase, Amazon GameLift Servers sta tentando una serie di attività e testando elementi chiave, tra cui la fattibilità del server di gioco, le impostazioni di configurazione del runtime e la capacità del server di gioco di connettersi al Amazon GameLift Servers servizio tramite Server SDK.

Note

In questa fase, puoi accedere in remoto a un'istanza del parco istanze per esaminare ulteriormente i problemi. Per informazioni, consulta [Connect a flotte di istanze](#).

I problemi possibili includono:

- I processi del server non iniziano a funzionare. Ciò suggerisce un problema con le impostazioni di configurazione del runtime del parco macchine (eventi `FLEET_VALIDATION_LAUNCH_PATH_NOT_FOUND` o `FLEET_VALIDATION_EXECUTABLE_RUNTIME_FAILURE`. Verifica di aver impostato correttamente il percorso di lancio e i parametri di lancio opzionali.
- I processi del server iniziano a funzionare, ma la flotta non riesce ad attivarsi. Se i processi del server vengono avviati ed eseguiti correttamente, ma il parco macchine non passa allo stato Attivo, è probabile che il processo del server non riesca a comunicare con il Amazon GameLift Servers servizio. Verifica che il server di gioco stia effettuando le chiamate SDK del server corrette (vedi [Inizializza il processo del server](#)):
 - Il processo del server non riesce a inizializzare (evento `SERVER_PROCESS_SDK_INITIALIZATION_TIMEOUT`). Il processo server non sta chiamando `InitSdk()` correttamente.
 - Il processo del server non riesce a notificare Amazon GameLift Servers quando è pronto per ospitare una sessione di gioco (evento `SERVER_PROCESS_PROCESS_READY_TIMEOUT`). Il processo del server è stato inizializzato ma la chiamata non è stata effettuata `ProcessReady()` in tempo.
- Una richiesta di connessione peering VPC non è riuscita. Per i parchi istanze creati con una connessione peering VPC (consulta [Per configurare il peering VPC con un nuovo parco istanze](#)), la connessione peering VPC viene effettuata durante la fase Activating (In fase di attivazione). Se una connessione peering VPC non riesce per qualsiasi motivo, il nuovo parco istanze non potrà passare allo stato Active (Attivo). È possibile monitorare l'esito positivo o

negativo della richiesta di peering chiamando. [describe-vpc-peering-connections](#) Assicurati di verificare che esista un'autorizzazione di peering VPC valida ([describe-vpc-peering-authorizations](#), poiché le autorizzazioni sono valide solo per 24 ore.

Problemi dei processi del server

I processi del server si avviano, ma si interrompono dopo breve tempo o sono poco stabili.

Oltre a problemi con la build del gioco, questo problema si può verificare quanto tenti di eseguire nell'istanza troppi processi del server contemporaneamente. Il numero ottimale di processi contemporanei dipende dal tipo di istanza e dai requisiti di risorse del server di gioco. Prova a ridurre il numero di processi contemporanei, impostati nella configurazione di runtime del parco istanze, per vedere se vi è un miglioramento delle prestazioni. Puoi modificare la configurazione di runtime di una flotta utilizzando la Amazon GameLift Servers console (modifica le impostazioni di allocazione della capacità del parco veicoli) o chiamando il comando AWS CLI [update-runtime-configuration](#)

Problemi di eliminazione del parco istanze

Non è possibile terminare il parco istanze a causa del conteggio istanze massimo.

Il messaggio di errore indica che il parco istanze in fase di eliminazione dispone ancora di istanze attive, situazione non consentita. È prima necessario ridimensionare il parco istanze a zero istanze attive. Questa operazione viene eseguita impostando manualmente il conteggio istanze desiderato del parco istanze su "0" e quindi attendendo che il dimensionamento abbia effetto. Assicurati di disattivare il ridimensionamento automatico, che neutralizzerà le impostazioni manuali.

Le azioni VPC non sono autorizzate.

Questo problema si applica solo alle flotte per le quali sono state create specificamente connessioni peering VPC (vedi. [Configurare il peering VPC per Amazon GameLift Servers](#)). Questo scenario si verifica perché il processo di eliminazione di una flotta include anche l'eliminazione del VPC della flotta e di eventuali connessioni peering VPC. È necessario prima ottenere un'autorizzazione chiamando l'API del servizio for Amazon GameLift Servers [CreateVpcPeeringAuthorization\(\)](#) o utilizzando il comando AWS CLI. `create-vpc-peering-authorization` Una volta ottenuta l'autorizzazione, è possibile eliminare il parco istanze.

Accesso remoto alle risorse della flotta

Puoi connetterti a qualsiasi istanza delle tue flotte di container gestite o EC2 attive Amazon GameLift Servers. Per le flotte di container, puoi anche connetterti a singoli contenitori in esecuzione su un'istanza per risolvere le sessioni di gioco, ispezionare i log ed eseguire il debug dei problemi di runtime. I motivi più comuni per accedere in remoto a un'istanza includono:

- Risolvi i problemi relativi all'integrazione del server di gioco.
- Perfeziona la configurazione di runtime e altre impostazioni specifiche del parco veicoli.
- Monitora l'attività dei server di gioco in tempo reale, ad esempio il tracciamento dei log.
- Esegui strumenti di benchmarking utilizzando il traffico effettivo dei giocatori.
- Esamina problemi specifici relativi a una sessione di gioco o a un processo del server.

Quando ti connetti a un'istanza, tieni presente quanto segue:

- Puoi connetterti a qualsiasi istanza in un parco istanze attivo. In genere, non è possibile connettersi a flotte non attive, ad esempio flotte in fase di attivazione o in stato di errore. (Queste flotte potrebbero avere una disponibilità limitata per un breve periodo di tempo.) Per assistenza con i problemi di attivazione della flotta, consulta [Debug dei problemi del parco istanze di Amazon GameLift Servers](#).
- La connessione a un'istanza attiva non influisce sull'attività di hosting dell'istanza. L'istanza continua ad avviare e arrestare i processi del server in base alla configurazione di runtime. Attiva ed esegue sessioni di gioco. L'istanza potrebbe chiudersi in risposta a un evento di ridimensionamento o a un altro evento.
- Qualsiasi modifica apportata ai file o alle impostazioni sull'istanza potrebbe influire sulle sessioni di gioco attive dell'istanza e sui giocatori connessi.

Connect a flotte di istanze

Puoi connetterti in remoto a qualsiasi istanza EC2 Amazon GameLift Servers gestita attiva o gestita di una flotta di container per risolvere i problemi dei server di gioco, ispezionare i log ed eseguire il debug del comportamento di runtime. Connect tramite la Amazon GameLift Servers console o la AWS CLI.

Accesso remoto tramite la console

Puoi connetterti alle istanze del parco istanze direttamente dalla Amazon GameLift Servers console utilizzando Amazon EC2 Systems Manager (SSM). Questo metodo fornisce un accesso sicuro senza richiedere configurazioni o gestione delle credenziali aggiuntive. Per le flotte di container, dopo la connessione all'istanza, è possibile accedere ai singoli contenitori in esecuzione su di essa. Per ulteriori informazioni, consulta [Connect ai container](#).

1. Nella Amazon GameLift Servers console, scegli Managed EC2 o Managed containers dal pannello di navigazione, quindi Fleets.
2. Scegli l'ID della flotta che contiene l'istanza a cui desideri accedere.
3. Nella pagina dei dettagli del parco istanze, scegli la scheda Istanze per visualizzare tutte le istanze di calcolo del parco istanze.
4. Seleziona l'istanza a cui desideri connetterti, quindi scegli Connetti. Viene visualizzata la finestra di dialogo Connetti all'istanza che fornisce informazioni sui dettagli della connessione e consente di visualizzare lo script che verrà utilizzato per connettersi all'istanza. Conferma selezionando nuovamente Connect.
5. Nella finestra di dialogo di connessione, scegli Esegui per creare una nuova sessione SSM. Il sistema autentica la sessione tramite AWS Key Management Service (AWS KMS) e apre un terminale nel browser.

Note

L'accesso remoto basato sulla console è disponibile per le flotte che eseguono la versione 5.x del server SDK. Per le flotte che eseguono versioni SDK precedenti, utilizzate il metodo AWS CLI descritto nella sezione seguente.

Accesso remoto con la AWS CLI

Le seguenti istruzioni descrivono come connettersi in remoto a un'istanza utilizzando l'interfaccia a riga di AWS comando (CLI). [Puoi anche effettuare chiamate programmatiche utilizzando l' AWS SDK, come documentato nel riferimento all'API di servizio per. Amazon GameLift Servers](#)

Raccogli dati delle istanze

Per connetterti a un'istanza Amazon GameLift Servers gestita di un parco istanze EC2, hai bisogno delle seguenti informazioni:

- L'ID dell'istanza a cui desideri connetterti. È possibile utilizzare l'ID dell'istanza o l'ARN.
- L'SDK del server per la Amazon GameLift Servers versione utilizzata sull'istanza. L'SDK del server è integrato con la build del gioco in esecuzione sull'istanza.

Le seguenti istruzioni descrivono come completare queste attività utilizzando la AWS CLI. Devi conoscere l'ID della flotta per l'istanza a cui desideri connetterti.

1. Ottieni il nome del computer. Ottieni un elenco di tutti i computer attivi della flotta. [Calcola l'elenco](#) delle chiamate con un Fleet ID o ARN. Per una flotta con una sola sede, specifica solo l'identificatore della flotta. Per una flotta con più sedi, specifica l'identificatore della flotta e una posizione. Con le flotte EC2 gestite, `list-compute` restituisce un elenco di istanze del parco istanze e la proprietà `ComputeName` è l'ID dell'istanza. Trova l'elaborazione a cui desideri accedere.

Richiesta

```
aws gamelift list-compute \  
  --fleet-id fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa \  
  --location sa-east-1
```

Risposta

```
{  
  "ComputeList": [  
    {  
      "FleetId": "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",  
      "FleetArn": "arn:aws:gamelift:us-west-2::fleet/  
fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",  
      "ComputeName": "i-0abc12d3e45fa6b78",  
      "IpAddress": "00.00.000.00",  
      "DnsName":  
"b08444ki909kvqu6zpw3is24x5pyz4b6m05i3jbxvpk9craztu0lqrbbrbnbkks.uwp57060n1k6dn1nw49b78hg1  
west-2.amazongamelift.com",  
      "ComputeStatus": "Active",  
      "Location": "sa-east-1",
```

```

    "CreationTime": "2023-07-09T22:51:45.931000-07:00",
    "OperatingSystem": "AMAZON_LINUX_2023",
    "Type": "c4.large"
  }
]
}
```

2. Trova la versione SDK del server. Per queste informazioni devi cercare la build implementata nella flotta. La versione Server SDK è una proprietà di compilazione.
 - a. Chiama [describe-fleet-attributes](#) con un ID flotta o un ARN per ottenere l'ID di build e l'ARN della flotta.
 - b. Chiama [describe-build](#) con l'ID di build o l'ARN per ottenere la versione SDK del server della build.

Esempio:

Richiesta

```
aws gamelift describe-fleet-attributes \
  --fleet-ids fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa
```

Risposta

```
{
  "FleetAttributes": [
    {
      "FleetId": "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
      "ComputeType": "EC2",
      "BuildId": "build-3333cccc-44dd-55ee-66ff-00001111aa22",
      . . .
    }
  ]
}
```

Richiesta

```
aws gamelift describe-build \
  --build-id build-3333cccc-44dd-55ee-66ff-00001111aa22
```

Risposta

```
"Build": {  
  "BuildId": "build-1111aaaa-22bb-33cc-44dd-5555eeee66ff",  
  "Name": "My_Game_Server_Build_One",  
  "OperatingSystem": "AMAZON_LINUX_2023",  
  "ServerSdkVersion": "5.1.1",  
  . . .  
}
```

Connect a un'istanza (server SDK 5)

Se l'istanza a cui desideri connetterti esegue una build di gioco con server SDK versione 5.x, connettiti all'istanza utilizzando Amazon EC2 Systems Manager (SSM). Puoi accedere alle istanze remote in esecuzione su Windows o Linux. Per le flotte di container, dopo la connessione all'istanza, puoi accedere ai singoli container in esecuzione su di essa. Per ulteriori informazioni, consulta [Connect ai container](#).

Prima di iniziare:

Completa i passaggi di configurazione SSM e installa il plug-in SSM sul tuo computer locale. Per ulteriori informazioni, consulta [Configurazione di SSM](#) e [Installazione del plug-in Session Manager per la AWS CLI](#) nella Guida per l'utente di Amazon EC2 Systems Manager.

1. Richiedere le credenziali di accesso all'istanza. Chiama [get-compute-access](#) con l'ID della flotta e il nome di calcolo dell'istanza a cui desideri connetterti. Amazon GameLift Servers restituisce un set di credenziali temporanee per accedere all'istanza. Esempio:

Richiesta

```
aws gamelift get-compute-access \  
--compute-name i-11111111a222b333c \  
--fleet-id fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa \  
--region us-west-2
```

Risposta

```
{
  "ComputeName": "i-11111111a222b333c",
  "Credentials": {
    "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",
    "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
    "SessionToken": "AQoDYXdzEJr...<remainder of session token>"
  },
  "FleetArn": "arn:aws:gamelift:us-west-2::fleet/fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
  "FleetId": "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa"
}
```

2. Esporta le credenziali di accesso (opzionale). È possibile esportare le credenziali in variabili di ambiente e utilizzarle per configurare la AWS CLI per l'utente predefinito. Per maggiori dettagli, consulta [Variabili di ambiente per configurare la AWS CLI nella Guida](#) per l' AWS Command Line Interface utente.

```
export AWS_ACCESS_KEY_ID=ASIAIOSFODNN7EXAMPLE
export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
export AWS_SESSION_TOKEN=AQoDYXdzEJr...<remainder of session token>
```

3. Connect all'istanza fleet. Avvia una sessione SSM con l'istanza a cui desideri connetterti. Includi la AWS regione o la posizione dell'istanza. Per ulteriori informazioni, tra cui come configurare SSM e il plug-in SSM, consulta [Avvio di una sessione \(AWS CLI\)](#) nella Guida per l'utente di Amazon EC2 Systems Manager.

La richiesta di avvio della sessione utilizzerà automaticamente le credenziali acquisite nella fase 1.

```
aws ssm start-session \
--target i-11111111a222b333c \
--region us-west-2 \
```

Note

Se viene visualizzato un errore di accesso negato, è possibile che una variabile di `AWS_PROFILE` ambiente sia impostata AWS su un profilo, il che fa sì che AWS CLI utilizzi le credenziali errate per l'accesso remoto. Per risolvere, disattiva temporaneamente la variabile di ambiente `AWS_PROFILE`. In alternativa, è possibile

creare un AWS profilo personalizzato per le credenziali di accesso remoto e aggiungere il parametro della riga di `--profile` comando alla `richiestastart-session`.

Connect a un'istanza (server SDK 4.x o precedente)

Se l'istanza a cui vuoi connetterti esegue una build di gioco con server SDK versione 4 o precedente, usa le seguenti istruzioni. Puoi connetterti a istanze che eseguono Windows o Linux. Connect a un'istanza Windows utilizzando un client RDP (Remote Desktop Protocol). Connect a un'istanza Linux utilizzando un client SSH.

1. Richiedere le credenziali di accesso all'istanza. Se disponi di un ID di istanza, usa il comando [get-instance-access](#) per richiedere le credenziali di accesso. Se la richiesta va a buon fine, Amazon GameLift Servers restituisce il sistema operativo, l'indirizzo IP e un insieme di credenziali (nome utente e chiave segreta) dell'istanza. Il formato delle credenziali dipende dal sistema operativo dell'istanza. Utilizzare le istruzioni seguenti per recuperare le credenziali per RDP o SSH.
 - Per le istanze Windows: per connettersi a un'istanza Windows, RDP richiede un nome utente e una password. La richiesta `get-instance-access` restituisce tali valori come stringhe semplici, pertanto è possibile utilizzare i valori restituiti così come sono. Credenziali di esempio:

```
"Credentials": {
  "Secret": "aA1bBB2cCCd3EEE",
  "UserName": "gl-user-remote"
}
```

- Per le istanze Linux: per connettersi a un'istanza Linux, SSH richiede un nome utente e una chiave privata. Amazon GameLift Server mette chiavi private RSA e le restituisce come una singola stringa, con il carattere newline (`\n`) che indica le interruzioni di riga. Per rendere utilizzabile la chiave privata, procedi nel seguente modo: (1) converti la stringa in un `.pem` file e (2) imposta le autorizzazioni per il nuovo file. Credenziali di esempio restituite:

```
"Credentials": {
  "Secret": "-----BEGIN RSA PRIVATE KEY-----
nEXAMPLEKEYKCAQEay7WZhaDsrA1W3mR1QtvhwyORRX8gnxgDAfRt/gx42kWXsT4rXE/b5CpSgie/
\nvBoU7jLxx92pNHoFnByP+Dc21eyyz6CvjTmWA0JwfWiW5/akH7i05dSrvC7dQkW2duV5QuUdE0QW
\nZ/aNxMniGQE6XAgfwlnXVBwrerrQo+ZWQeqiUwwMkuEbLeJFLhMCvYURpUMSC1oehm449i1x9X1F
-----"
```

```

\nG50TCFe0zf18dqqCP6GzbPaIjiU19xX/az0R9V+tpU0zEL+wmXnZt3/nHPQ5xvD20JH67km6SuPW
\noPzev/D8V+x4+bHthfSjR9Y7DvQFjfbVwHXigBdtZcU2/wei8D/HYwIDAQABAOIBAGZ1kaEvnqrqu
\n/uler7vgIn5m7lN5LKw4hJLAIW6tUT/fzvtcHK0SkbQCQXuriHmQ2MQyJX/0kn2NfjLV/
ufGxbL1\nmb5qwmGUnEpJaZD6QSSs3kICLwUUYUiGfc0uiSbmJoap/
GTLU0W5Mfcv36PaBUNy5p53V6G7hXb2\nbahyWyJNfjLe4M86yd2YK3V2CmK+X/
B0sShnJ36+hjrXPPWmV3N9zEmCdJjA+K15DYmhm/
tJWSD9\n81oGk9TopEp7CkIfatEATyyZiVqoRq6k64iuM9JkA30zdXzMqexXVJ1TLZVEH0E7bh1Y9d801ozR
\noQs/FiZNAx2iijCWyv0lpjE73+kCgYEA9mZtyhkHkFDpwrSM1APaL8oNAbbjwEy7Z5Mqfq1
+lIp1\nYkriL0DbLXlvRAH+yHPRit2hH0jtUNZh4Axv+cpq09qbUI3+43eEy24B7G/Uh
+GTfbjsXs0xQx/x\np9otyVwc7hsQ5TA5PZb
+mvkJ50BEKzet9XcKw0NBYELGhnEPe7cCgYEA06Vgov6YHleHui9kHuws
\nayav0elc5zkxjF9nfHFJRry21R1trw2Vdpn+9g481URrpzWV0Eihvm+xTtmaZ1Sp//lkq75XDwnU
\nWA8gkn603QE3fq2yN98BURsAKdJfJ5RL1HvGQvTe10HLYYXpJnEkHv+Unl2ajLivWUt5pbBrKbUC
\nngYBjb0+0Zk0sCcpZ29sbzjYjpIddErySIyRX5gV2uNQwAjLdp9PfN295yQ+BxMBXiIycWVQiw0bH
\nnoMo7yykABY70zd5wQewBQ4AdSlWSX4nGDtsiFxiI5sKuAAe0CbTosy1s8w8fxoJ5Tz1sdoxNeGs
\nArq6Wv/G16zQuAE9zK9vvwKBgF+09VI/1wJBirsDGz9whVwFFPrTkJNvJZzYt69qezx1sjgFKshy
\nWBhd4xHZtmCqpBP1AymEjr/T01bxyARMXMnIOWIANXMG4KGSyl1mzSVAoQ+fqr+cJ3d0dyP11j
\nnjjb0Ed/NY8frlNDxAVHE8BSkdsx2f6ELEyBKJSRr9snRAoGAMrTwYneXzvTskF/S5Fyu0i0egLda
\nNWUH38v/nDCgEpIXD5Hn3qAEcju1IjmbwlvtW+nY2jVhv7UGd8MjwUTNGItbdb6nsYqM2asrnF3qS
\nVRkAKKKYeGjKpUfVTrW0YFjXkfcrR/V+QFL50ndHAKJXjW7a4ejJLncTzmZSpYzwApc=\n-----END
RSA PRIVATE KEY-----",
    "UserName": "gl-user-remote"
}

```

Quando si utilizza la AWS CLI, è possibile generare automaticamente un `.pem` file includendo i parametri `--query` e `--output` nella richiesta. `get-instance-access`

Per impostare le autorizzazioni sul file `.pem`, eseguire il comando seguente:

```
$ chmod 400 MyPrivateKey.pem
```

2. Aprire una porta per la connessione remota. È possibile accedere alle istanze delle Amazon GameLift Servers flotte tramite qualsiasi porta autorizzata nella configurazione del parco veicoli. È possibile visualizzare le impostazioni di porta del parco istanze tramite il comando [describe-fleet-port-settings](#).

Come best practice, si consiglia di aprire le porte per l'accesso remoto solo quando necessario e di chiuderle al termine dell'operazione. Non puoi aggiornare le impostazioni delle porte dopo aver creato una flotta ma prima che sia attiva. Se rimani bloccato, ricrea la flotta con le impostazioni del porto aperte.

Utilizzare il comando [update-fleet-port-settings](#) per aggiungere un'impostazione di porta per la connessione remota (ad esempio 22 per SSH o 3389 per RDP). Per il valore di intervallo IP, specificare gli indirizzi IP dei dispositivi che si intende utilizzare per la connessione (convertiti in formato CIDR). Esempio:

```
$ AWS gamelift update-fleet-port-settings
--fleet-id "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa"
--inbound-permission-authorizations
"FromPort=22,ToPort=22,IpRange=54.186.139.221/32,Protocol=TCP"
```

L'esempio seguente apre la porta 3389 su una flotta di Windows

```
$ AWS gamelift update-fleet-port-settings
--fleet-id "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa"
--inbound-permission-authorizations
"FromPort=3389,ToPort=3389,IpRange=54.186.139.221/32,Protocol=TCP"
```

3. Aprire un client di connessione remota. Utilizzare Desktop remoto per le istanze Windows o SSH per le istanze Linux. Connettersi all'istanza utilizzando l'indirizzo IP, l'impostazione di porta e le credenziali di accesso.

Esempio SSH:

```
ssh -i MyPrivateKey.pem gl-user-remote@192.0.2.0
```

Visualizza i file su istanze remote

Una volta connesso a un'istanza del parco istanze, hai accesso utente e amministrativo completo. Ciò significa che hai anche la possibilità di causare errori o guasti con l'hosting dei giochi. Se l'istanza ospita giochi con giocatori attivi, potresti correre il rischio di interrompere le sessioni di gioco e far cadere i giocatori, oppure di interrompere i processi di chiusura del gioco, il che potrebbe causare errori nei dati e nei registri di gioco salvati.

Per le flotte di container, dopo l'accesso al container (vedi [Connect ai container](#)), le seguenti posizioni di file si applicano all'interno del filesystem del contenitore. Per le flotte EC2, queste posizioni di file si applicano direttamente all'istanza.

Cerca queste risorse su un'istanza di hosting:

- File della build di gioco. Questi file sono la build del gioco su cui hai caricato Amazon GameLift Servers. Includono uno o più file eseguibili, risorse e dipendenze del server di gioco. I file di build del gioco si trovano in una directory principale chiamata: game
 - Su Windows: c:\game
 - In Linux: /local/game
- File di log del gioco. Trova i file di registro generati dal tuo server di gioco nella directory game principale in qualsiasi percorso di directory che hai designato.
- Risorse di hosting di Amazon GameLift Servers. La directory principale `Whitewater` contiene i file utilizzati dal Amazon GameLift Servers servizio per gestire l'attività di hosting dei giochi. Non modificate questi file per nessun motivo.
- Configurazione di runtime. Non accedete alla configurazione di runtime per le singole istanze. Per apportare modifiche a una proprietà di configurazione di runtime, aggiorna la configurazione di runtime della flotta (vedi il funzionamento dell' AWS SDK [UpdateRuntimeConfiguration](#) o il AWS CLI [update-runtime-configuration](#)).
- Dati sulla flotta. Un file JSON contiene informazioni sulla flotta a cui appartiene l'istanza, utilizzabili dai processi del server in esecuzione sull'istanza. Il file JSON si trova nella seguente posizione:
 - Su Windows: C:\GameMetadata\gamelift-metadata.json
 - In Linux: /local/gamemetadata/gamelift-metadata.json
- Certificati TLS. Se l'istanza fa parte di una flotta in cui è abilitata la generazione di certificati TLS, cerca i file di certificato, tra cui il certificato, la catena di certificati, la chiave privata e il certificato principale nella seguente posizione:
 - Su Windows: c:\\GameMetadata\Certificates
 - In Linux: /local/gamemetadata/certificates/

Connect ai container

Per le flotte di Amazon GameLift Servers container, puoi accedere ai container dei server di gioco in esecuzione su un'istanza della flotta. Usa l'accesso ai container per risolvere i problemi delle sessioni di gioco, controllare i log ed eseguire il debug dei problemi di runtime.

Connect a un container

Prima di iniziare:

Connect all'istanza fleet. Per istruzioni, consulta [Connect a flotte di istanze](#).

Esegui il comando seguente per elencare i contenitori in esecuzione sull'istanza:

```
sudo docker ps
```

L'output elenca tutti i contenitori in esecuzione sull'istanza, inclusi i contenitori del server di gioco e Amazon GameLift Servers i contenitori interni. Cerca contenitori con l'immagine del tuo server di gioco per identificare i contenitori del server di gioco.

Output di esempio:

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS
b9676e9489f5	game-server-container	"/bin/sh -c ./ \$GAME_..."	2 days ago	Up 2 days
1d1c8443efe2	support-container	"/bin/sh -c ./ \$SUPPO..."	2 days ago	Up 2 days

Per connetterti a un contenitore del server di gioco, usa l'ID breve del contenitore nella CONTAINER ID colonna. Questo ti dà l'accesso completo in lettura e scrittura al filesystem del contenitore.

```
sudo docker exec -it container-short-id sh
```

Connect a un container tramite la console

Puoi connetterti ai contenitori dei server di gioco dalla Amazon GameLift Servers console utilizzando Amazon EC2 Systems Manager (SSM). Questo metodo fornisce un accesso sicuro senza richiedere configurazioni o gestione delle credenziali aggiuntive. Puoi connetterti a un container dalla scheda Computer o dalla scheda Sessioni di gioco nella pagina dei dettagli della flotta.

1. Nella Amazon GameLift Servers console, scegli Contenitori gestiti dal pannello di navigazione, quindi Fleets.
2. Scegli l'ID della flotta che contiene il container o la sessione di gioco a cui desideri accedere.
3. Nella pagina dei dettagli della flotta, scegli una delle seguenti schede:

- **Calcola:** elenca i container in circolazione nella flotta. Seleziona il contenitore a cui desideri connetterti.
 - **Sessioni di gioco:** elenca le sessioni di gioco per la flotta. Seleziona la sessione di gioco per connetterti al contenitore che la ospita.
4. Scegli **Connetti**. Copia il comando visualizzato per connetterti al contenitore, quindi scegli nuovamente **Connetti**.
 5. Nella finestra di dialogo di connessione, scegli **Esegui** per creare una nuova sessione SSM. Il sistema autentica la sessione tramite AWS Key Management Service (AWS KMS) e apre un terminale nel browser.
 6. Dopo esserti connesso all'istanza, incolla il comando docker dal passaggio 4 ed esegilo sull'istanza per accedere al contenitore.

Connect al container che ospita una sessione di gioco

Per connetterti al contenitore del server di gioco che ospita una sessione di gioco specifica, segui questi passaggi.

1. Ottieni il nome del computer. Chiama [describe-game-sessions](#) per ricevere il programma `ComputeName` per la sessione di gioco.

Richiesta

```
aws gamelift describe-game-sessions \
  --fleet-id fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa \
  --game-session-id arn:aws:gamelift:us-west-2::gamesession/
fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa/gs-1111aaaa-2222-3333-4444-5555bbbb66cc
```

Risposta

```
{
  "GameSessions": [
    {
      "GameSessionId": "arn:aws:gamelift:us-west-2::gamesession/
fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa/
gs-1111aaaa-2222-3333-4444-5555bbbb66cc",
      "FleetId": "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
      "ComputeName": "62c5ff7f7a9a445d84877074c80aeafc",
      "Status": "ACTIVE",
```

```

    . . .
  }
]
}

```

Annota il `ComputeName` valore della risposta (ad esempio, `62c5ff7f7a9a445d84877074c80aeafc`).

2. Ottieni l'accesso al calcolo e gli attributi del contenitore. Chiama [get-compute-access](#) con l'ID della flotta e il nome del calcolo.

La risposta include i seguenti campi:

- **ContainerIdentifiers**— La `ContainerName` e `ContainerRuntimeId` per ogni contenitore.
- **GameServerContainerGroupDefinitionArn**— L'ARN della definizione del gruppo di contenitori.
- **Credentials**— Credenziali temporanee per connettersi all'istanza.

Richiesta

```

aws gamelift get-compute-access \
  --fleet-id fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa \
  --compute-name 62c5ff7f7a9a445d84877074c80aeafc

```

Risposta

```

{
  "ComputeName": "62c5ff7f7a9a445d84877074c80aeafc",
  "ContainerIdentifiers": [
    {
      "ContainerName": "game-server",
      "ContainerRuntimeId":
"02accb92cd9bef3373300e7151d5c2b3dcca3b06eff1bb4e345085fc008d4678"
    }
  ],
  "Credentials": {
    "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",
    "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY",
    "SessionToken": "AQoDYXdzEJr...<remainder of session token>"
  }
}

```

```

    },
    "FleetId": "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
    "GameServerContainerGroupDefinitionArn": "arn:aws:gamelift:us-
west-2::containergroupdefinition/MyGameServerGroup"
  }
}

```

3. Collegati all'istanza. Utilizza le credenziali del passaggio 2 per connetterti all'istanza della flotta. Per istruzioni dettagliate, vedi [Connect a flotte di istanze](#).
4. Trova il nome del contenitore del server di gioco. Chiama [describe-container-group-definition](#) utilizzando il GameServerContainerGroupDefinitionArn passaggio 2 per identificare il nome del contenitore del server di gioco.

Richiesta

```

aws gamelift describe-container-group-definition \
  --name arn:aws:gamelift:us-west-2::containergroupdefinition/MyGameServerGroup

```

Risposta

```

{
  "ContainerGroupDefinition": {
    "ContainerGroupDefinitionArn": "arn:aws:gamelift:us-
west-2:123456789012:containergroupdefinition/MyGameServerGroup:3",
    "Name": "MyGameServerGroup",
    "ContainerGroupType": "GAME_SERVER",
    "GameServerContainerDefinition": {
      "ContainerName": "game-server",
      . . .
    },
    . . .
  }
}

```

Nota il GameServerContainerDefinition.ContainerName valore (ad esempio, game-server).

5. Identifica l'ID di runtime del contenitore del server di gioco. Utilizzando il nome del contenitore del server di gioco del passaggio precedente, trova la ContainerIdentifiers voce corrispondente nella get-compute-access risposta del passaggio 2. Annotare il valore ContainerRuntimeId.

6. Connect al container. Usa `ContainerRuntimeId` come ID del contenitore ed esegui il seguente comando:

```
sudo docker exec -it  
02accb92cd9bef3373300e7151d5c2b3dcca3b06eff1bb4e345085fc008d4678 sh
```

Visualizza le mappature delle porte del contenitore

Le mappature delle porte mostrano come le porte dei container vengono mappate alle porte di connessione delle istanze della flotta. A ogni porta container che accetta traffico in entrata viene assegnata una porta di connessione sull'istanza. Puoi controllare le mappature delle porte per scoprire quali porte di connessione sono mappate alle porte del container o per risolvere i problemi di connessione. Puoi visualizzare le mappature delle porte nella Amazon GameLift Servers console o utilizzare l'SDK o. AWS CLI AWS

Visualizza le mappature delle porte nella console

Nella Amazon GameLift Servers console, scegli Contenitori gestiti dal pannello di navigazione, quindi Flotte. Scegli una flotta per aprire la pagina dei dettagli della flotta. Puoi visualizzare le mappature dei porti nelle seguenti posizioni:

- Pagina dei dettagli dell'istanza: mostra le mappature delle porte per il gruppo di contenitori per istanza sull'istanza selezionata.
- Pagina dei dettagli di calcolo: mostra le mappature delle porte per il gruppo di contenitori del server di gioco sul computer selezionato.

Entrambe le pagine includono un campo di ricerca per filtrare le mappature delle porte in base al nome del contenitore.

Visualizza le mappature delle porte con il AWS CLI

Utilizzate il `describe-container-group-port-mappings` comando per recuperare le mappature delle porte per un gruppo di container su un'istanza della flotta. Specificate il tipo di gruppo di contenitori e un `--compute-name` (per i gruppi di server di gioco) o un `--instance-id` (per i gruppi di istanze). Facoltativamente, includi il `--container-name` parametro per filtrare i risultati in un contenitore specifico.

Esempio: gruppo di contenitori del server di gioco

Il comando seguente recupera le mappature delle porte per il gruppo di contenitori del server di gioco su un computer specifico.

```
aws gamelift describe-container-group-port-mappings \
  --fleet-id fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa \
  --container-group-type GAME_SERVER \
  --compute-name 62c5ff7f7a9a445d84877074c80aeafc
```

In caso di successo, Amazon GameLift Servers restituisce una risposta come la seguente:

```
{
  "FleetId": "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
  "Location": "us-west-2",
  "ContainerGroupDefinitionArn": "arn:aws:gamelift:us-west-2:123456789012:containergroupdefinition/MyGameServerGroup",
  "ContainerGroupType": "GAME_SERVER",
  "ComputeName": "62c5ff7f7a9a445d84877074c80aeafc",
  "InstanceId": "i-1234567890abcdef0",
  "ContainerGroupPortMappings": [
    {
      "ContainerName": "MyGameServer",
      "ContainerRuntimeId": "a1b2c3d4e5f6",
      "ContainerPortMappings": [
        {
          "ContainerPort": 7777,
          "ConnectionPort": 1025,
          "Protocol": "UDP"
        },
        {
          "ContainerPort": 8080,
          "ConnectionPort": 1026,
          "Protocol": "TCP"
        }
      ]
    }
  ]
}
```

Esempio: gruppo di contenitori per istanza

Il comando seguente recupera le mappature delle porte per il gruppo di contenitori per istanza su un'istanza specifica.

```
aws gamelift describe-container-group-port-mappings \  
  --fleet-id fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa \  
  --container-group-type PER_INSTANCE \  
  --instance-id i-1234567890abcdef0
```

In caso di successo, Amazon GameLift Servers restituisce una risposta simile alla seguente:

```
{  
  "FleetId": "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",  
  "Location": "us-west-2",  
  "ContainerGroupDefinitionArn": "arn:aws:gamelift:us-  
west-2:123456789012:containergroupdefinition/MyPerInstanceGroup",  
  "ContainerGroupType": "PER_INSTANCE",  
  "InstanceId": "i-1234567890abcdef0",  
  "ContainerGroupPortMappings": [  
    {  
      "ContainerName": "MySupportContainer",  
      "ContainerRuntimeId": "f6e5d4c3b2a1",  
      "ContainerPortMappings": [  
        {  
          "ContainerPort": 8443,  
          "ConnectionPort": 2025,  
          "Protocol": "TCP"  
        }  
      ]  
    }  
  ]  
}
```

Scalabilità della capacità di hosting di giochi con Amazon GameLift Servers

La capacità di hosting, misurata in istanze, rappresenta il numero di sessioni di gioco che Amazon GameLift Servers possono ospitare contemporaneamente e il numero di giocatori simultanei che tali sessioni di gioco possono ospitare. Una delle attività più impegnative dell'hosting di giochi è scalare la capacità per soddisfare la domanda dei giocatori senza sprecare denaro in risorse di cui non hai bisogno.

La capacità viene regolata a livello di ubicazione della flotta. Tutte le flotte hanno almeno una sede: la AWS regione di origine della flotta. Quando si visualizza o si aumenta la capacità, le informazioni vengono elencate per località, inclusa la regione di origine della flotta e qualsiasi località remota aggiuntiva.

Puoi impostare manualmente il numero di istanze da mantenere oppure configurare il dimensionamento automatico per regolare dinamicamente la capacità in base alle variazioni della domanda dei giocatori. Ti consigliamo di iniziare attivando l'opzione di ridimensionamento automatico basato sulla destinazione. L'obiettivo del dimensionamento automatico basato sugli obiettivi è mantenere risorse di hosting sufficienti per soddisfare i giocatori attuali, più un po' di spazio aggiuntivo per gestire picchi imprevisti nella domanda dei giocatori. Per la maggior parte dei giochi, il dimensionamento automatico basato sugli obiettivi offre una soluzione di dimensionamento estremamente efficace.

Puoi eseguire la maggior parte delle attività di ridimensionamento del parco veicoli utilizzando la console. Amazon GameLift Servers Puoi anche utilizzare un AWS SDK o il AWS Command Line Interface (AWS CLI) con l'[API del servizio](#) per. Amazon GameLift Servers

Argomenti

- [Per gestire la capacità della flotta nella console](#)
- [Imposta i limiti Amazon GameLift Servers di capacità](#)
- [Imposta manualmente la capacità per una Amazon GameLift Servers flotta](#)
- [Capacità di dimensionamento automatico del parco istanze con Amazon GameLift Servers](#)
- [Gestire la scalabilità e azzerare Amazon GameLift Servers la flotta To/From](#)
- [Flotte di Amazon GameLift Servers container in scala](#)

Per gestire la capacità della flotta nella console

1. Apri la [Amazon GameLift Servers console](#).
2. Nel riquadro di navigazione, scegli Hosting, Fleets.
3. Nella pagina Flotte, scegli il nome di una flotta attiva per aprire la pagina dei dettagli della flotta.
4. Scegli la scheda Scaling. In questa scheda puoi:
 - Visualizza le metriche di scalabilità storiche per l'intera flotta.
 - Visualizza e aggiorna le impostazioni di capacità per ogni sede del parco veicoli, inclusi i limiti di scalabilità e le impostazioni di capacità correnti.
 - Aggiorna la scalabilità automatica basata sugli obiettivi, visualizza le politiche di scalabilità automatica basate su regole applicate all'intera flotta e sospendi l'attività di auto scaling per ogni sede.

Imposta i limiti Amazon GameLift Servers di capacità

Quando scalate la capacità di hosting per una sede Amazon GameLift Servers della flotta, manualmente o tramite scalabilità automatica, considerate i limiti di scalabilità della sede. Tutte le sedi del parco veicoli hanno un limite minimo e massimo che definisce l'intervallo consentito per la capacità della sede. Per impostazione predefinita, i limiti per le sedi del parco veicoli hanno un minimo di 0 istanze e un massimo di 1 istanza. Prima di poter scalare una sede del parco veicoli, modifica i limiti.

Se utilizzi la scalabilità automatica, il limite massimo consente di ampliare la posizione di una flotta Amazon GameLift Servers per soddisfare la domanda dei giocatori, ma evita costi di hosting eccessivi, come durante un attacco DDOS. Imposta un [CloudWatch allarme Amazon](#) per avvisarti quando la capacità si avvicina al limite massimo, in modo da poter valutare la situazione e regolarla manualmente secondo necessità. (Puoi anche [creare un allarme di fatturazione](#) per monitorare AWS i costi.) Il limite minimo è utile per mantenere la disponibilità dell'hosting, anche quando la domanda dei giocatori è bassa.

Puoi impostare i limiti di capacità per le sedi di una flotta nella [Amazon GameLift Serversconsole](#) o utilizzando AWS Command Line Interface (AWS CLI).

Per impostare i limiti di capacità

Console

1. Apri la [Amazon GameLift Servers console](#).
2. Nel pannello di navigazione, scegli Hosting, Fleets.
3. Nella pagina Flotte, scegli il nome di una flotta attiva per aprire la pagina dei dettagli della flotta.
4. Nella scheda Scalabilità, in Scalabilità della capacità, seleziona un'ubicazione della flotta, quindi scegli Modifica.
5. Nella finestra di dialogo Modifica capacità di scalabilità, imposta il numero di istanze per Dimensione minima, Istanze desiderate e Dimensione massima.
6. Scegli Conferma.

AWS CLI

1. Controllate le impostazioni di capacità correnti. In una finestra della riga di comando, usa il [describe-fleet-location-capacity](#) comando con l'ID della flotta e la posizione per cui desideri modificare la capacità. Questo comando restituisce un [FleetCapacity](#) oggetto che include le impostazioni di capacità correnti della posizione. Determina se i nuovi limiti di istanza possono soddisfare l'attuale impostazione delle istanze desiderate.

```
aws gamelift describe-fleet-location-capacity \
  --fleet-id <fleet identifier> \
  --location <location name>
```

2. Update limit settings (Aggiornare le impostazioni dei limiti). In una finestra della riga di comando, utilizzate il [update-fleet-capacity](#) comando con i seguenti parametri. È possibile regolare i limiti dell'istanza e il conteggio desiderato delle istanze con lo stesso comando.

```
--fleet-id <fleet identifier>
--location <location name>
--max-size <maximum capacity for scaling>
--min-size <minimum capacity for scaling>
--desired-instances <fleet capacity goal>
```

Esempio:

```
aws gamelift update-fleet-capacity \  
  --fleet-id fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa \  
  --location us-west-2 \  
  --max-size 10 \  
  --min-size 1 \  
  --desired-instances 10
```

Se la richiesta ha esito positivo, Amazon GameLift Servers restituisce l'ID della flotta. Se il nuovo `min-size` valore `max-size` o il valore è in conflitto con l'`desired-instances` impostazione corrente, Amazon GameLift Servers restituisce un errore.

Imposta manualmente la capacità per una Amazon GameLift Servers flotta

Quando crei un nuovo parco, Amazon GameLift Servers imposta automaticamente le istanze desiderate su un'istanza in ogni posizione del parco. Quindi, Amazon GameLift Servers implementa una nuova istanza in ogni posizione. Per modificare la capacità della flotta, puoi aggiungere una politica di scalabilità automatica basata sull'obiettivo oppure puoi impostare manualmente il numero di istanze che desideri per una sede.

L'impostazione manuale della capacità di una flotta può essere utile quando non è necessaria la scalabilità automatica o quando è necessario mantenere la capacità a un livello specifico. L'impostazione manuale della capacità funziona solo se non si utilizza una politica di auto scaling basata sulla destinazione. Se disponi di una politica di scalabilità automatica basata sulla destinazione, ripristina immediatamente la capacità desiderata in base alle proprie regole di scalabilità.

È possibile impostare manualmente la capacità nella Amazon GameLift Servers console o utilizzando (). AWS Command Line Interface AWS CLI Lo stato della flotta deve essere attivo.

Sospensione del dimensionamento automatico

Puoi sospendere tutte le attività di dimensionamento automatico per ciascuna posizione del parco istanze. Con la sospensione del dimensionamento automatico, il numero desiderato di istanze nella posizione del parco istanze rimane invariato a meno che non venga modificato manualmente.

Quando si sospende il dimensionamento automatico di una posizione, ciò influisce sulle policy attuali del parco istanze e su quelle che potresti definire in futuro.

Per impostare manualmente la capacità del parco istanze

Console

1. Apri la [Amazon GameLift Servers console](#).
2. Nel riquadro di navigazione, scegli Hosting, Fleets.
3. Nella pagina Flotte, scegli il nome di una flotta attiva per aprire la pagina dei dettagli della flotta.
4. Nella scheda Ridimensionamento, in Posizioni di ridimensionamento automatico sospese, seleziona ogni posizione per cui desideri sospendere la scalabilità automatica, quindi scegli Sospendi.
5. In Capacità di scalabilità, seleziona una posizione che desideri impostare manualmente, quindi scegli Modifica.
6. Nella finestra di dialogo Modifica capacità di scalabilità, imposta il valore preferito per Istanze desiderate, quindi scegli Conferma. Indica Amazon GameLift Servers il numero di istanze da mantenere attive, pronte per ospitare sessioni di gioco.

Amazon GameLift Servers risponde alle modifiche distribuendo istanze aggiuntive o chiudendo quelle non necessarie. Al termine di questo processo, il numero di istanze attive nella posizione cambia per corrispondere al valore aggiornato delle istanze desiderate. Questo processo può richiedere alcuni istanti.

AWS CLI

1. Controllate le impostazioni di capacità correnti. In una finestra della riga di comando, usa il [describe-fleet-location-capacity](#) comando con l'ID della flotta e la posizione per cui desideri modificare la capacità. Questo comando restituisce un [FleetCapacity](#) oggetto che include le impostazioni di capacità correnti della posizione. Determina se i limiti delle istanze possono soddisfare la nuova impostazione delle istanze desiderata.

```
aws gamelift describe-fleet-location-capacity \
  --fleet-id <fleet identifier> \
  --location <location name>
```


2. Aggiornare la capacità desiderata. Utilizza il [update-fleet-capacity](#) comando con l'ID della flotta, la posizione e un nuovo valore per le istanze desiderate. Se questo valore non rientra nell'intervallo limite corrente, puoi regolare i valori limite con lo stesso comando.

```
--fleet-id <fleet identifier>
--location <location name>
--desired-instances <fleet capacity as an integer>
--max-size <maximum capacity> [Optional]
--min-size <minimum capacity> [Optional]
```

Esempio:

```
aws gamelift update-fleet-capacity \
  --fleet-id fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa \
  --location us-west-2 \
  --desired-instances 5 \
  --max-size 10 \
  --min-size 1
```

Se la richiesta ha esito positivo, Amazon GameLift Servers restituisce l'ID della flotta. Se la nuova impostazione delle istanze desiderate non rientra nei limiti minimo e massimo, Amazon GameLift Servers restituisce un errore.

Capacità di dimensionamento automatico del parco istanze con Amazon GameLift Servers

Usa la scalabilità automatica per Amazon GameLift Servers scalare dinamicamente la capacità della tua flotta in risposta all'attività dei server di gioco. Man mano che i giocatori arrivano e iniziano le sessioni di gioco, la scalabilità automatica può aggiungere altre istanze; man mano che la domanda dei giocatori diminuisce, la scalabilità automatica può terminare le istanze non necessarie. La scalabilità automatica è un modo efficace per ridurre al minimo le risorse e i costi di hosting, garantendo al contempo un'esperienza di gioco fluida e veloce.

Per utilizzare la scalabilità automatica, si creano politiche di scalabilità che indicano Amazon GameLift Servers quando aumentare o diminuire. Esistono due tipi di politiche di scalabilità: basate sugli obiettivi e basate su regole. L'approccio basato sugli obiettivi, il monitoraggio degli obiettivi, è una soluzione completa. La consigliamo come opzione più semplice ed efficace. Le politiche di

scalabilità basate su regole richiedono la definizione di ogni aspetto del processo decisionale relativo alla scalabilità automatica, utile per risolvere problemi specifici. Questa soluzione funziona al meglio come supplemento alla scalabilità automatica basata sull'obiettivo.

Puoi gestire la scalabilità automatica basata sulla destinazione utilizzando la Amazon GameLift Servers console, AWS Command Line Interface (AWS CLI) o un SDK. AWS È possibile gestire la scalabilità automatica basata su regole utilizzando solo l'SDK AWS CLI o un AWS SDK, sebbene sia possibile visualizzare le politiche di scalabilità basate su regole nella console.

Argomenti

- [Dimensionamento automatico basato sugli obiettivi](#)
- [Dimensionamento automatico con policy basate su regole](#)

Dimensionamento automatico basato sugli obiettivi

Scalabilità automatica basata sull'obiettivo per Amazon GameLift Servers regolare i livelli di capacità in base alla metrica della flotta. `PercentAvailableGameSessions` Questo parametro rappresenta il buffer disponibile del parco istanze per gli aumenti improvvisi della domanda di giocatori.

Il motivo principale per mantenere un buffer di capacità è il tempo di attesa di un giocatore. Quando gli slot per le sessioni di gioco sono pronti e in attesa, bastano pochi secondi per coinvolgere nuovi giocatori nelle sessioni di gioco. Se non ci sono risorse disponibili, i giocatori devono attendere la fine delle sessioni di gioco esistenti o la disponibilità di nuove risorse. L'avvio di nuove istanze e processi del server può richiedere alcuni minuti.

Quando configuri la scalabilità automatica basata sull'obiettivo, specifica la dimensione del buffer che desideri che il parco veicoli gestisca. Poiché `PercentAvailableGameSessions` misura la percentuale di risorse disponibili, la dimensione effettiva del buffer è una percentuale della capacità totale del parco veicoli. Amazon GameLift Servers aggiunge o rimuove istanze per mantenere la dimensione del buffer di destinazione. Con un buffer di grandi dimensioni, riduci al minimo i tempi di attesa, ma paghi anche per risorse extra che potresti non utilizzare. Se i giocatori sono più tolleranti rispetto ai tempi di attesa, è possibile ridurre i costi impostando un piccolo buffer.

Per impostare la scalabilità automatica basata sull'obiettivo

Console

1. Apri la [Amazon GameLift Servers console](#).

2. Nel riquadro di navigazione, scegli Hosting, Fleets.
3. Nella pagina Flotte, scegli il nome di una flotta attiva per aprire la pagina dei dettagli della flotta.
4. Scegli la scheda Scaling. Questa scheda mostra i parametri del dimensionamento storico del parco istanze e contiene i controlli per modificare le impostazioni di dimensionamento correnti.
5. In Capacità di scalabilità, verifica che i limiti di dimensione minima e dimensione massima siano appropriati per il parco veicoli. Con la scalabilità automatica abilitata, la capacità si regola tra questi due limiti.
6. Nella politica di scalabilità automatica basata su Target, scegli Modifica.
7. Nella finestra di dialogo Modifica politica di autoscaling basata sull'obiettivo, per Percentuale di sessioni di gioco disponibili, imposta la percentuale che desideri mantenere, quindi scegli Conferma. Dopo aver confermato le impostazioni, Amazon GameLift Servers aggiunge una nuova politica basata sulla destinazione in Politica di scalabilità automatica basata su Target.

AWS CLI

1. Impostare i limiti di capacità. Imposta i valori limite utilizzando il comando. [update-fleet-capacity](#) Per ulteriori informazioni, consulta [Imposta i limiti Amazon GameLift Servers di capacità](#).
2. Creare una nuova policy. Apri una finestra della riga di comando e usa il [put-scaling-policy](#) comando con le impostazioni dei parametri della tua politica. Per aggiornare una policy esistente, specificare il nome della policy e fornire una versione completa della policy aggiornata.

```
--fleet-id <unique fleet identifier>
--name "<unique policy name>"
--policy-type <target- or rule-based policy>
--metric-name <name of metric>
--target-configuration <buffer size>
```

Esempio:

```
aws gamelift put-scaling-policy \
  --fleet-id "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa" \
  --name "My_Target_Policy_1" \
  --policy-type "TargetBased" \
```

```
--metric-name "PercentAvailableGameSessions" \  
--target-configuration "TargetValue=5"
```

Dimensionamento automatico con policy basate su regole

Le politiche di scalabilità basate su regole Amazon GameLift Servers forniscono un controllo preciso quando si ridimensiona automaticamente la capacità di una flotta in risposta all'attività dei giocatori. Per ogni policy, puoi collegare la scalabilità a una delle diverse metriche della flotta, identificare un punto di attivazione e personalizzare l'evento di scalabilità verticale o discendente che risponde.

[Le politiche basate su regole sono utili per integrare la scalabilità basata sugli obiettivi per gestire circostanze speciali.](#)

Una politica basata su regole stabilisce quanto segue: «Se una metrica della flotta soddisfa o supera un valore di soglia per un certo periodo di tempo, modifica la capacità della flotta di un importo specificato». Questo argomento descrive la sintassi utilizzata per un'istruzione di policy e offre assistenza nella creazione e nella gestione di policy basate su regole.

Gestire policy basate su regole

[Crea, aggiorna o elimina politiche basate su regole utilizzando un AWS SDK o AWS Command Line Interface \(\)AWS CLI con l'API di servizio per. Amazon GameLift Servers](#) Le policy attive possono essere visualizzate nella console Amazon GameLift Servers.

Per interrompere temporaneamente tutte le politiche di scalabilità per una flotta, usa il comando. AWS CLI [stop-fleet-actions](#)

Per creare o aggiornare una politica di scalabilità basata su regole ():AWS CLI

1. Impostare i limiti di capacità. Imposta uno o entrambi i valori limite utilizzando il comando. [update-fleet-capacity](#) Per ulteriori informazioni, consulta [Imposta i limiti Amazon GameLift Servers di capacità.](#)
2. Creare una nuova policy. Apri una finestra della riga di comando e usa il [put-scaling-policy](#) comando con le impostazioni dei parametri della tua politica. Per aggiornare una policy esistente, specificare il nome della policy e fornire una versione completa della policy aggiornata.

```
--fleet-id <unique fleet identifier>  
--name "<unique policy name>"  
--policy-type <target- or rule-based policy>  
--metric-name <name of metric>
```

```
--comparison-operator <comparison operator>
--threshold <threshold integer value>
--evaluation-periods <number of minutes>
--scaling-adjustment-type <adjustment type>
--scaling-adjustment <adjustment amount>
```

Esempio:

```
aws gamelift put-scaling-policy \
  --fleet-id fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa \
  --name "Scale up when AGS<50" \
  --policy-type RuleBased \
  --metric-name AvailableGameSessions \
  --comparison-operator LessThanThreshold \
  --threshold 50 \
  --evaluation-periods 10 \
  --scaling-adjustment-type ChangeInCapacity \
  --scaling-adjustment 1
```

Per eliminare una politica di scalabilità basata su regole utilizzando: AWS CLI

- Aprire una finestra della riga di comando e utilizzare il [delete-scaling-policy](#) comando con l'ID della flotta e il nome della politica.

Esempio:

```
aws gamelift delete-scaling-policy \
  --fleet-id fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa \
  --name "Scale up when AGS<50"
```

Sintassi per le regole di ridimensionamento automatico

Per creare una dichiarazione di politica di scalabilità basata su regole, specifica sei variabili:

Se *<metric name>* rimane *<comparison operator>* *<threshold value>* per *<evaluation period>*, modifica la capacità della flotta utilizzando. *<adjustment type>* to/by *<adjustment value>*

Ad esempio, questa dichiarazione politica avvia un evento di ampliamento ogni volta che la capacità aggiuntiva di una flotta è inferiore a quella necessaria per gestire 50 nuove sessioni di gioco:

Se `AvailableGameSessions` rimane a less than 50 per 10 minutes, utilizzare `ChangeInCapacity` per modificare la capacità del parco istanze di 1 instances.

Nome parametro

Per avviare un evento di scalabilità, collega una politica di scalabilità automatica a una delle seguenti metriche specifiche del parco veicoli. Per le descrizioni complete delle metriche, consulta [Parametri di Amazon GameLift Servers per parchi istanze](#)

- Attivazione delle sessioni di gioco
- Sessioni di gioco attive
- Sessioni di gioco disponibili
- Percentuale di sessioni di gioco disponibili
- Istanze attive
- Sessioni giocatore disponibili
- Sessioni giocatore correnti
- Istanze inattive
- Percentuale di istanze inattive

Se la flotta è in coda per una sessione di gioco, puoi utilizzare le seguenti metriche:

- Profondità della coda: il numero di richieste di sessioni di gioco in sospeso per cui questa flotta è la migliore posizione di hosting disponibile.
- Tempo di attesa: tempo di attesa specifico per la flotta. L'attesa necessaria per il completamento della richiesta di sessione di gioco in attesa meno recente. Il tempo di attesa di un parco istanze corrisponde al tempo in coda della richiesta corrente meno recente.

Operatore di confronto

Indica Amazon GameLift Servers come confrontare i dati della metrica con il valore di soglia. Gli operatori di confronto validi includono maggiore di ($>$), minore di ($<$), greater than or equal ($>=$) e minore o uguale ($<=$).

Valore di soglia

Quando il valore metrico specificato raggiunge o supera il valore di soglia, avvia un evento di scala. Questo valore è sempre un numero intero positivo.

Periodo di valutazione

La metrica deve soddisfare o superare il valore di soglia per l'intera durata del periodo di valutazione prima di iniziare un evento di scalabilità. La lunghezza del periodo di valutazione è consecutiva: se il parametro scende al di sotto del valore di soglia, il periodo di valutazione inizia nuovamente.

Tipo e valore di adeguamento

Questo set di variabili collabora per Amazon GameLift Servers specificare come regolare la capacità della flotta all'inizio di un evento di scalabilità. Scegli tra tre possibili tipi di regolazione:

- **Modifica della capacità:** aumenta o diminuisce la capacità attuale di un numero specifico di istanze. Il valore di adeguamento deve essere impostato sul numero di istanze da aggiungere o rimuovere dal parco istanze. I valori positivi aggiungono istanze, mentre quelli negativi le rimuovono. Ad esempio, un valore di «-10" riduce il parco istanze di 10 istanze, indipendentemente dalla dimensione totale del parco istanze.
- **Variazione percentuale della capacità:** aumenta o diminuisce la capacità attuale di una percentuale specificata. Imposta il valore di regolazione sulla percentuale di cui desideri aumentare o diminuire la capacità della flotta. I valori positivi aggiungono istanze, mentre quelli negativi le rimuovono. Ad esempio, per un parco istanze con 50 istanze, una variazione percentuale di «20" aggiunge 10 istanze al parco istanze.
- **Capacità esatta:** aumenta o diminuisce la capacità attuale fino a un valore specifico. Il valore di adeguamento deve essere impostato sul numero esatto di istanze che si desidera mantenere nel parco istanze.

Suggerimenti per il ridimensionamento automatico basato su regole

I seguenti suggerimenti possono aiutarti a ottenere il massimo dalla scalabilità automatica con policy basate su regole.

Utilizzare più policy

Puoi avere più politiche di scalabilità automatica per una flotta contemporaneamente. Lo scenario più comune consiste in una policy mirata che gestisce la maggior parte delle esigenze di dimensionamento e utilizza le policy basate su regole per gestire i casi limite. Non ci sono limiti all'utilizzo di più politiche.

Con più policy, ogni policy si comporta in modo indipendente. Non è possibile controllare la sequenza degli eventi di scalabilità. Ad esempio, se hai più politiche che favoriscono la scalabilità, è possibile

che l'attività dei giocatori dia inizio a più eventi di scalabilità contemporaneamente. Evita politiche che si avviano l'una con l'altra. Ad esempio, è possibile creare un ciclo infinito se si creano policy scalabili verso l'alto e verso il basso che impostano la capacità oltre la soglia reciproca.

Impostare la capacità minima e massima

Ogni parco istanze ha un limite di capacità massimo e uno minimo. Questa funzionalità è importante quando si utilizza il ridimensionamento automatico. La scalabilità automatica non imposta mai la capacità su un valore al di fuori di questo intervallo. Per impostazione predefinita, i parchi istanze di nuova creazione hanno come limite minimo 0 e come limite massimo 1. Affinché la politica di scalabilità automatica influisca sulla capacità come previsto, aumentate il valore massimo.

La capacità del parco veicoli è inoltre limitata dai limiti relativi al tipo di istanza del parco macchine e dalle quote di servizio previste. Account AWS Non puoi impostare un minimo e un massimo al di fuori di questi limiti e quote di account.

Monitorare i parametri dopo una modifica della capacità

Dopo aver modificato la capacità in risposta a una policy di auto scaling, Amazon GameLift Servers attende 10 minuti prima di rispondere ai trigger della stessa policy. Questa Amazon GameLift Servers attesa consente di aggiungere nuove istanze, avviare i server di gioco, connettere i giocatori e iniziare a raccogliere dati dalle nuove istanze. Durante questo periodo, Amazon GameLift Servers valuta la policy rispetto alla metrica e tiene traccia del periodo di valutazione della policy, che riprende dopo che si verifica un evento di scalabilità. Ciò significa che una politica di scalabilità potrebbe avviare un altro evento di scalabilità subito dopo il termine del tempo di attesa.

Non vi è alcun tempo di attesa tra gli eventi di scalabilità che iniziano le diverse politiche di auto scaling.

Gestire la scalabilità e azzerare Amazon GameLift Servers la flotta To/From

Amazon GameLift Servers supporta il ridimensionamento automatico da e verso zero istanze in base all'attività della sessione di gioco. Questa opzione di capacità gestita consente alle sedi della tua flotta di scalare fino a zero istanze dopo un determinato periodo di assenza di attività di gioco e di ridimensionarsi automaticamente quando vengono richieste sessioni di gioco.

La scalabilità da e verso zero istanze offre diversi vantaggi:

- Ottimizzazione dei costi: elimina i costi di elaborazione durante i periodi di inattività eseguendo zero istanze in assenza di attività durante le sessioni di gioco.
- Scalabilità orizzontale reattiva automatica: le posizioni delle flotte si ridimensionano automaticamente su un'istanza quando viene richiesta una sessione di gioco, eliminando la necessità di un intervento manuale.
- Gestione semplificata: non è necessario regolare manualmente la capacità della flotta in base alla domanda prevista dei giocatori o to/from alle esigenze di sviluppo.

Quando abiliti Scale To/From Zero su una flotta, Amazon GameLift Servers monitora l'attività delle sessioni di gioco e regola automaticamente la capacità della flotta:

- Scalabilità a zero: dopo un periodo configurato senza attività di sessioni di gioco, Amazon GameLift Servers ridimensiona la posizione della flotta fino a zero istanze.
- Scalabilità orizzontale da zero: quando viene ricevuta una richiesta di creazione di una sessione di gioco, Amazon GameLift Servers ridimensiona la posizione della flotta a un'istanza, permettendo la ripresa della scalabilità automatica.
- Scalabilità continua: dopo la scalabilità orizzontale, la flotta riprende a utilizzare politiche di scalabilità automatica configurate per gestire la capacità.

Comportamento di scalabilità

Amazon GameLift Servers avvia il processo di scalabilità per una sede della flotta dopo che è trascorso il periodo di inattività configurato senza alcuna attività della sessione di gioco. Questo è definito come un periodo in cui:

- Non ci sono sessioni di gioco attive nella sede della Flotta.
- Non è stata fatta alcuna richiesta per creare nuove sessioni di gioco nella sede della Flotta.

Durante lo scale-in, Amazon GameLift Servers imposterà a zero la capacità minima e desiderata per l'ubicazione della flotta, ampliandola rapidamente per ridurre i costi.

Comportamento di scalabilità orizzontale

Quando viene ricevuta una richiesta di creazione di una sessione di gioco mentre la posizione Fleet è a zero istanze:

- Amazon GameLift Servers avvia immediatamente la scalabilità orizzontale di un'istanza.
- Il tentativo di piazzamento della sessione di gioco può continuare con altre postazioni delle flotte o delle flotte, a seconda della configurazione delle code (se utilizzate).

Note

La scalabilità orizzontale da zero richiede tempo per il provisioning e l'inizializzazione delle istanze. I giocatori potrebbero riscontrare tempi di attesa più lunghi per la prima sessione di gioco dopo un periodo di inattività. Per questo motivo, questa funzione si abbina al meglio a Fleets Queues con più sedi. and/or

To/From Configurazione di Scale Zero

Scaling To/From Zero viene configurato aggiornando una flotta esistente.

Console

1. Apri la [Amazon GameLift Servers console](#).
2. Nel pannello di navigazione, scegli Hosting, Fleets.
3. Nella pagina Flotte, scegli il nome di una flotta attiva per aprire la pagina dei dettagli della flotta.
4. In Scalabilità, seleziona ogni posizione per la quale desideri configurare la scala to/from zero, quindi scegli Modifica.
5. Nella finestra di dialogo Modifica capacità di scalabilità, seleziona «Automatico» per Strategia di capacità minima, imposta il valore preferito in minuti per Imposta la capacità minima su 0 dopo, quindi scegli Conferma.

Amazon GameLift Servers ridurrà quindi le postazioni della flotta selezionate a 0 istanze dopo che non vi sarà stata alcuna attività di sessione di gioco per la durata configurata. Successivamente, quando viene effettuata una richiesta per una sessione di gioco in questa località, Amazon GameLift Servers verrà ridimensionata un'istanza il più rapidamente possibile. Questo processo richiederà un po' di tempo.

AWS CLI

- Configura la scala to/from zero. In una finestra della riga di comando, usa il [update-fleet-capacity](#) comando con l'ID della flotta, la posizione e la configurazione della capacità gestita per configurare la scala to/from zero.

```
aws gamelift update-fleet-capacity \
    --fleet-id <fleet identifier> \
    --location <location name> \
    --managed-capacity-configuration
ScaleInAfterInactivityMinutes=60,ZeroCapacityStrategy=SCALE_TO_AND_FROM_ZERO
```

Esempio:

```
aws gamelift update-fleet-capacity \
    --fleet-id
fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa \
    --location us-west-2 \
    --desired-instances 5 \
    --max-size 10 \
    --managed-capacity-configuration
ScaleInAfterInactivityMinutes=60,ZeroCapacityStrategy=SCALE_TO_AND_FROM_ZERO
```

Se la richiesta ha esito positivo, Amazon GameLift Servers restituisce la configurazione della capacità aggiornata. Amazon GameLift Servers ridimensionerà quindi le postazioni della flotta selezionate a 0 istanze dopo che non vi sarà stata alcuna attività di sessione di gioco per la durata configurata. Successivamente, quando viene effettuata una richiesta per una sessione di gioco in questa località, Amazon GameLift Servers verrà ridimensionata un'istanza il più rapidamente possibile. Questo processo richiederà un po' di tempo.

Best practice

Considerate i seguenti consigli quando utilizzate Scale To/From Zero:

- Imposta i periodi di inattività appropriati: bilancia i risparmi sui costi con la frequenza dei cicli di scale-in/scale-out. I tempi di attesa più brevi per la scalabilità a zero possono massimizzare i risparmi, ma comporterebbero avviamenti a freddo più frequenti.
- Utilizzalo con carichi di lavoro prevedibili: Scale To/From Zero è ideale per giochi con chiari periodi di inattività, come development/test ambienti o giochi con orari diversi non di punta.
- Monitora le CloudWatch metriche: monitora gli eventi di ridimensionamento della flotta e i tempi di posizionamento delle sessioni di gioco per ottimizzare la configurazione.
- Combinazione con politiche di scalabilità: utilizza Scale To/From Zero insieme alla scalabilità automatica basata su obiettivi o regole per una gestione completa della capacità.

Flotte di Amazon GameLift Servers container in scala

Una delle attività più impegnative dell'hosting di giochi è scalare la capacità per soddisfare la domanda dei giocatori senza sprecare denaro in risorse di cui non hai bisogno. In una flotta di container gestita, puoi scalare la capacità della flotta aggiungendo o rimuovendo istanze della flotta.

Quando crei una nuova flotta, Amazon GameLift Servers imposta la capacità desiderata della flotta su un'istanza e distribuisce un'istanza nella regione di origine della flotta. Per un parco veicoli con più sedi, Amazon GameLift Servers distribuisce un'istanza nella regione di origine e in ogni postazione remota. Una volta raggiunto lo stato della flotta ACTIVE, è possibile aumentare la capacità desiderata per aumentare o ridurre la capacità desiderata di ridimensionamento.

Puoi utilizzare le funzionalità Amazon GameLift Servers di scalabilità per modificare la capacità manualmente o impostare la scalabilità automatica in base alla domanda dei giocatori:

- Imposta il ridimensionamento automatico con tracciamento del target. Per informazioni, consulta [Dimensionamento automatico basato sugli obiettivi](#).
- Modifica manualmente la capacità della tua flotta. Per informazioni, consulta [Imposta manualmente la capacità per una Amazon GameLift Servers flotta](#).

Quando ridimensionate una flotta di container, considerate l'impatto dell'aggiunta o della rimozione di istanze sulla capacità della flotta di ospitare sessioni di gioco e giocatori.

- Sessioni di gioco per istanza
 - Ogni processo del server di gioco in esecuzione su un'istanza rappresenta la capacità di ospitare una sessione di gioco.

- Usa questa formula per calcolare il numero di sessioni di gioco eseguite contemporaneamente su un'istanza di una flotta di container:

```
[Game sessions per instance] = [# of game server processes per game server container] * [# of game server container groups per instance]
```

Se l'architettura del contenitore esegue contemporaneamente un processo del server di gioco nel contenitore del server di gioco, le sessioni di gioco per istanza equivalgono al numero di gruppi di contenitori di server di gioco per istanza.

- Per i gruppi di contenitori di server di gioco per esempio, chiama [DescribeContainerFleet](#) per ottenere il `MaximumGameServerContainerGroupsPerInstance` valore `GameServerContainerGroupsPerInstance` or.
- Il numero di gruppi di contenitori di server di gioco che rientrano in un'istanza dipende dalla quantità di memoria disponibile dopo aver Amazon GameLift Servers riservato una parte per il sistema operativo e i servizi di supporto. Per i dettagli su come viene calcolata la memoria disponibile e una tabella di riferimento dei tipi di istanze più comuni, consulta [Comprendi l'allocazione della memoria della flotta di container](#).
- Giocatori per istanza
 - Sei tu a decidere il numero di slot per i giocatori da consentire in ogni sessione di gioco. A seconda di come la tua soluzione di hosting gestisce il posizionamento delle sessioni di gioco, puoi definire i giocatori per sessione di gioco nella configurazione di matchmaking o nelle chiamate per avviare una sessione di gioco.
 - Usa questa formula per calcolare il numero di giocatori che possono giocare contemporaneamente alla tua partita su un'istanza di flotta di container:

```
[Players per instance] = [# of game sessions per instance] * [# of player slots per game session]
```

Per ottenere l'attuale capacità totale di una flotta di container, chiama [DescribeFleetCapacity](#) o [DescribeFleetLocation Capacity](#) per ottenere il numero di gruppi di container di server di gioco presenti nella flotta. I gruppi attivi sono quelli che attualmente ospitano sessioni di gioco. I gruppi inattivi sono pronti a ospitare una nuova sessione di gioco. Moltiplica questi valori per il numero di processi server per gruppo di contenitori di server di gioco.

Monitoraggio di Amazon GameLift Servers

Il monitoraggio è un elemento importante per mantenere l'affidabilità, la disponibilità e le prestazioni delle Amazon GameLift Servers e altre AWS soluzioni. Esistono tre impieghi principali per i parametri di Amazon GameLift Servers: monitorare l'integrità del sistema e impostare gli allarmi, tenere traccia delle prestazioni del server di gioco e gestire la capacità utilizzando il dimensionamento automatico o manuale.

AWS fornisce i seguenti strumenti di monitoraggio per osservare Amazon GameLift Servers, segnalare quando qualcosa non va e intraprendere azioni automatiche quando necessario:

- **Amazon GameLift Servers console:** utilizza l'interfaccia grafica per gestire le Amazon GameLift Servers risorse e tenere traccia delle attività di hosting dei giochi.
- **Metriche di telemetria del server:** ti Amazon GameLift Servers consentono di emettere metriche personalizzate direttamente dai tuoi server di gioco tramite SDK e integrazione di plug-in. Puoi definire e monitorare le tue metriche specifiche del gioco insieme ai dati integrati su prestazioni, rete, memoria e tempistica. Tutte le metriche possono essere pubblicate su [Amazon Managed Service for Prometheus](#) e monitorate in Amazon Grafana utilizzando dashboard completamente personalizzabili. Puoi anche pubblicare su Amazon CloudWatch per l'integrazione con altri AWS servizi. Il sistema di telemetria è completamente personalizzabile: puoi creare query Prometheus personalizzate per ricavare metriche aggiuntive oltre a quelle integrate. Per le guide all'implementazione specifiche per il tuo stack tecnologico, consulta [the section called "Monitoraggio con metriche di telemetria del server"](#)
- **Amazon CloudWatch:** puoi monitorare i Amazon GameLift Servers parametri in tempo reale, nonché i parametri per altre AWS risorse e applicazioni che esegui sui AWS servizi. Amazon CloudWatch offre una suite di funzionalità di monitoraggio, tra cui strumenti per creare dashboard personalizzati e la possibilità di impostare allarmi che notificano o intervengono quando una metrica raggiunge una soglia specificata.
- **AWS CloudTrail**— acquisisce tutte le chiamate API e gli eventi correlati effettuati da o per conto del tuo AWS account e per Amazon GameLift Servers altri servizi. AWS I dati vengono forniti come file di log a un bucket Amazon S3 specificato dall'utente. Puoi identificare quali utenti e account hanno effettuato le chiamate AWS, l'indirizzo IP di origine da cui sono state effettuate le chiamate e quando sono avvenute le chiamate.
- **Registri delle sessioni di gioco:** puoi inviare messaggi server personalizzati per le tue sessioni di gioco in file di registro archiviati in Amazon S3.

Confronto delle metriche tra le fonti di monitoraggio

Amazon GameLift Servers fornisce metriche attraverso tre fonti principali: le metriche Fleet Activity della Amazon GameLift Servers console, le metriche di telemetria del server e le metriche Amazon CloudWatch Amazon GameLift Servers. Comprendere la sovrapposizione e le funzionalità uniche di ciascuna fonte ti aiuta a scegliere l'approccio di monitoraggio giusto per le tue esigenze.

Disponibilità delle metriche per fonte

Le tabelle seguenti mostrano quali metriche sono disponibili in diverse fonti di monitoraggio, organizzate per categoria di metriche.

Parametri dell'istanza

Metriche a livello di istanza per il monitoraggio della capacità e dello stato della flotta:

Disponibilità delle metriche delle istanze

Metrica	Console	CloudWatch	Telemetria
Istanze attive	✓	✓	✓
Istanze inattive	✓	✓	✓*
Percentuale di istanze inattive	✓	✓	✓*
Istanze desiderate	✓	✓	
Numero massimo di istanze	✓	✓	
Numero minimo di istanze	✓	✓	
Utilizzo CPU		✓	✓
Ingresso/uscita di rete		✓	✓
Disk/Storage read/write		✓	✓
Interruzioni delle istanze	✓	✓	
Istanze riciclate	✓	✓	

Metrica	Console	CloudWatch	Telemetria
Istanze non sane sostituite		✓	

* Disponibile tramite query Prometheus personalizzate che utilizzano dati di telemetria.

Metriche del server di gioco

Metriche relative ai processi del server e alle sessioni di gioco:

Disponibilità delle metriche dei server di gioco

Metrica	Console	CloudWatch	Telemetria
Sessioni di gioco attive	✓	✓	✓*
Attivazione delle sessioni di gioco	✓	✓	
Sessioni di gioco disponibili	✓	✓	
Percentuale di sessioni di gioco disponibili	✓	✓	
Sessioni di gioco attivabili simultanee	✓	✓	
Interruzioni della sessione di gioco	✓	✓	
Processi attivi del server			✓
Server di gioco sani			✓
Sessioni di gioco interrotte			✓

* Disponibile tramite query Prometheus personalizzate che utilizzano dati di telemetria.

Metriche sull'utilizzo dei giocatori

Metriche relative alla sessione del giocatore e agli utenti simultanei:

Disponibilità delle metriche di utilizzo dei giocatori

Metrica	Console	CloudWatch	Telemetria
Sessioni giocatore correnti	✓		
Attivazioni delle sessioni Player	✓		
Utenti simultanei globali (CCU)			✓
Localizzazione degli utenti simultanei (CCU)			✓
Capacità di localizzazione			✓

Metriche della flotta di container

Metriche specifiche per i container (disponibili solo CloudWatch per le flotte di container):

Disponibilità dei parametri della flotta di container

Metrica	Console	CloudWatch	Telemetria
Gruppi di container attivi		✓	✓
Gruppi di contenitori inattivi		✓	✓
Utilizzo dei container CPU/Memory		✓	✓
Traffico di rete di container		✓	✓
Operazioni di storage in container		✓	✓

Scelta della giusta fonte di monitoraggio

Seleziona il tuo approccio di monitoraggio in base alle tue esigenze specifiche:

- Utilizza le metriche di telemetria dei server per monitorare i dati completi sulle prestazioni dei server di gioco ed emettere metriche personalizzate specifiche per il gioco. Queste metriche forniscono informazioni dettagliate sulle prestazioni del server, sull'attività di rete, sull'utilizzo della memoria e sui dati di temporizzazione per tutti i motori di gioco e i server. SDKs Puoi definire

metriche personalizzate per gli eventi di gioco, le prestazioni della logica aziendale e i punti dati specifici dell'applicazione. Tutte le dashboard sono completamente personalizzabili e puoi creare query Prometheus personalizzate per ricavare metriche aggiuntive dai dati raccolti. Per ulteriori informazioni, consulta [the section called “Monitoraggio con metriche di telemetria del server”](#).

- Utilizza la Amazon GameLift Servers console per la gestione della flotta, la pianificazione della capacità e la supervisione operativa generale. La console offre una visione integrata dello stato della flotta e dell'attività dei giocatori. Per ulteriori informazioni, consulta [the section called “Gestisci le risorse di hosting dei giochi”](#).
- Usa Amazon CloudWatch per il monitoraggio automatico, gli avvisi e l'integrazione con altri AWS servizi. CloudWatch abilita dashboard personalizzati e automazione basata sugli allarmi. Per ulteriori informazioni, consulta [the section called “Monitora con CloudWatch”](#).
- Usa [Amazon Managed Service for Prometheus per](#) la raccolta e l'archiviazione di metriche ad alte prestazioni con funzionalità di interrogazione PromQL. Prometheus offre un'archiviazione scalabile di serie temporali per le metriche di telemetria dei server.
- Usa [Amazon Managed Grafana](#) per una visualizzazione avanzata e una dashboard completamente personalizzabile. Grafana offre GameLift dashboard predefinite che puoi personalizzare ed estendere, oltre a supportare la creazione di dashboard completamente personalizzate con più fonti di dati. Puoi creare query e visualizzazioni personalizzate per tenere traccia di tutte le metriche importanti per il tuo gioco.
- Utilizza più fonti per un monitoraggio completo. Combina la supervisione della console, le metriche di telemetria dei server e gli avvisi automatici per una visibilità completa sulla tua infrastruttura di hosting dei giochi.

Per funzionalità di monitoraggio aggiuntive, puoi anche utilizzare:

- [the section called “Registrazione di chiamate API”](#)— Tieni traccia delle chiamate API e degli eventi correlati per il controllo e la conformità.
- [the section called “Registrazione dei messaggi del server”](#)— Acquisisci messaggi personalizzati sul server e registri delle sessioni di gioco.

Argomenti

- [the section called “Gestisci le risorse di hosting dei giochi”](#)
- [the section called “Monitora con CloudWatch”](#)
- [the section called “Monitoraggio con metriche di telemetria del server”](#)

- [the section called “Registrazione di chiamate API”](#)
- [the section called “Registrazione dei messaggi del server”](#)

Monitora Amazon GameLift Servers con Amazon CloudWatch

Puoi monitorare Amazon GameLift Servers utilizzando Amazon CloudWatch, un AWS servizio che raccoglie dati grezzi e li elabora in metriche leggibili quasi in tempo reale. Queste statistiche vengono conservate per un periodo di 15 mesi per fornire una prospettiva storica delle prestazioni del server di gioco che ospita Amazon GameLift Servers. Puoi impostare allarmi che controllano determinate soglie e inviare notifiche o intraprendere azioni quando queste soglie vengono raggiunte. Per ulteriori informazioni, consulta la [Amazon CloudWatch User Guide](#).

[Per una migliore osservabilità lato server con metriche telemetriche dettagliate, Amazon GameLift Servers supporta anche il monitoraggio tramite Amazon Managed Service for Prometheus e Amazon Managed Grafana.](#) Queste metriche di telemetria forniscono dati dettagliati sulle prestazioni del server, tra cui:

- Orari del server: Server Delta Time (coerenza della frequenza di tick del server), Server Tick Rate (aggiornamenti al secondo), Server Tick Time (tempo di elaborazione per tick) e Server World Tick Time (tempo di aggiornamento del mondo di gioco)
- Metriche di rete: connessioni (connessioni di rete totali), rete I/O in byte e pacchetti (statistiche sul trasferimento dei dati) e perdita di pacchetti (percentuale di errori di trasmissione)
- Metriche della memoria: utilizzo della memoria in unità (consumo totale di memoria) e utilizzo della memoria fisica in percentuale (utilizzo della memoria)
- Utilizzo della CPU: percentuale di utilizzo della CPU (utilizzo delle risorse per processo del server di gioco)

Per le guide all'implementazione specifiche per il tuo SDK o plug-in, consulta. [the section called “Monitoraggio con metriche di telemetria del server”](#)

Le seguenti tabelle elencano i parametri e le dimensioni di Amazon GameLift Servers. Tutte le metriche disponibili in CloudWatch sono disponibili anche nella Amazon GameLift Servers console, che fornisce i dati sotto forma di set di grafici personalizzabili. Per accedere alle CloudWatch metriche dei tuoi giochi, usa l' Console di gestione AWS AWS CLI, o l'API. CloudWatch

Se una metrica non ha una posizione, utilizza la posizione principale.

Dimensioni per i parametri Amazon GameLift Servers

Amazon GameLift Servers supporta l'applicazione di filtri ai parametri per le seguenti dimensioni.

Dimensione	Description
Location	Filtra le metriche per una località di distribuzione del parco veicoli. Se una metrica non ha una posizione, utilizza la posizione principale.
FleetId	Filtra i parametri per un singolo parco istanze. Questa dimensione può essere utilizzata con tutti i parametri di parchi istanze per istanze, processi server, sessioni di gioco e sessioni giocatore.
MetricGroup	Filtra i parametri per una raccolta di parchi istanze. Aggiungi un parco veicoli a un gruppo di metriche aggiungendo il nome del gruppo di metriche agli attributi del parco veicoli (vedi UpdateFleetAttributes()). Questa dimensione può essere utilizzata con tutti i parametri di parchi istanze per istanze, processi server, sessioni di gioco e sessioni giocatore.
QueueName	Filtra i parametri per una singola coda. Questa dimensione viene utilizzata solo con i parametri per le code delle sessioni di gioco.
ConfigurationName	Filtra i parametri per una singola configurazione di abbinamento. Questa dimensione viene utilizzata solo con i parametri per le configurazioni di abbinamenti.
ConfigurationName-RuleName	Filtra i parametri per l'intersezione di una configurazione di abbinamento e una regola di abbinamento. Questa dimensione viene utilizzata solo con parametri per regole di abbinamento.

Dimensione	Description
InstanceType	Filtra i parametri per la designazione di un tipo di istanza EC2, come "c4.large". Questa dimensione viene utilizzata con parametri per istanze Spot.
OperatingSystem	Filtra i parametri per il sistema operativo di un'istanza. Questa dimensione viene utilizzata con i parametri per le istanze Spot.
GameServerGroup	Filtra i parametri FleetIQ per un gruppo di server di gioco.
ContainerGroupType	Filtra le metriche della flotta di container per distinguere tra le GameServer ContainerGroup metriche relative all'ubicazione della flotta e le PerInstance ContainerGroup metriche sulla posizione della flotta. Questa dimensione è inclusa per alcune metriche della flotta di container (ad esempio ContainerNetworkIn, ContainerStorageReadBytes) e consente di distinguere tra metriche per diversi tipi di gruppi di container.

Parametri di Amazon GameLift Servers per parchi istanze

Il namespace `AmazonGameLift` include i seguenti parametri relativi all'attività in un parco istanze o un gruppo di parchi istanze. I parchi istanze vengono utilizzati con una soluzione Amazon GameLift Servers gestita. Il Amazon GameLift Servers servizio invia metriche a CloudWatch ogni minuto.

Istanze

Metrica	Description
ActiveInstances	Istanze con stato ACTIVE (Attivo), ovvero istanze che stanno eseguendo processi server attivi. Il numero include le istanze inattive e quelle che ospitano una o più sessioni di gioco. Questo

Metrica	Description
	parametro misura la capacità di istanze totale corrente e può essere utilizzato con la scalabilità automatica. Unità: numero CloudWatch Statistiche pertinenti: media, minima, massima Dimensioni: Ubicazione
DesiredInstances	Il numero target di istanze attive che Amazon GameLift Servers sta elaborando per mantenerli e nel parco istanze. Con la scalabilità automatica questo valore viene determinato sulla base delle policy di scalabilità attualmente applicate. Senza scalabilità automatica, questo valore viene impostato manualmente. Questo parametro non è disponibile quando si visualizzano i dati per gruppi di parametri del parco istanze. Unità: numero CloudWatch Statistiche pertinenti: media, minima, massima Dimensioni: Ubicazione

Metrica	Description
IdleInstances	Istanze attive che ospitano attualmente zero (0) sessioni di gioco. Questo parametro misura la capacità disponibile ma non utilizzata. e può essere utilizzato con la scalabilità automatica. Unità: numero CloudWatch Statistiche pertinenti: media, minima, massima Dimensioni: Ubicazione
MaxInstances	Numero massimo di istanze consentite per il parco istanze. Tale numero determina la capacità massima durante la scalabilità automatica o manuale. Questo parametro non è disponibile quando si visualizzano i dati per gruppi di parametri del parco istanze. Unità: numero CloudWatch Statistiche pertinenti: media, minima, massima Dimensioni: Ubicazione
MinInstances	Numero minimo di istanze consentite per il parco istanze. Tale numero determina la capacità minima durante la scalabilità automatica o manuale. Questo parametro non è disponibile quando si visualizzano i dati per gruppi di parametri del parco istanze. Unità: numero CloudWatch Statistiche pertinenti: media, minima, massima Dimensioni: Ubicazione

Metrica	Description
PercentIdleInstances	Percentuale di tutte le istanze attive che sono inattive (calcolata come $\text{IdleInstances} / \text{ActiveInstances}$). Questo parametro può essere utilizzato per la scalabilità automatica. Unità: percentuale CloudWatch Statistiche pertinenti: media, minima, massima Dimensioni: Ubicazione
RecycledInstances	Numero di istanze spot che sono state riciclate e sostituite. Amazon GameLift Servers ricicla le istanze spot che attualmente non ospitano sessioni di gioco e che hanno un'alta probabilità di interruzione. Questa metrica è disponibile solo per le flotte EC2 gestite. Unità: numero CloudWatch Statistiche pertinenti: somma, media, minimo, massimo Dimensioni: Ubicazione

Metrica	Description
RecycledSpotInstances	Numero di istanze spot che sono state riciclate e sostituite. Amazon GameLift Servers ricicla le istanze spot che attualmente non ospitano sessioni di gioco e che hanno un'alta probabilità di interruzione. Questa metrica è disponibile solo per le flotte di container. Unità: numero CloudWatch Statistiche pertinenti: somma, media, minimo, massimo Dimensioni: Ubicazione
InstanceSpotInterruptions	Numero di istanze Spot che sono state interrotte. Unità: numero CloudWatch Statistiche pertinenti: somma, media, minima, massima Dimensioni: Ubicazione

Metrica	Description
CPUUtilization	Sistema metrico EC2. Amazon GameLift Servers Questa metrica rappresenta le prestazioni hardware di tutte le istanze attive in una flotta di sedi. La percentuale di tempo fisico della CPU utilizzata da Amazon EC2 per eseguire l'istanza, che include il tempo impiegato per eseguire sia il codice utente che il codice Amazon EC2. Gli strumenti del sistema operativo possono mostrare una percentuale diversa CloudWatch rispetto a fattori quali la simulazione di dispositivi legacy, la configurazione di dispositivi non legacy, i carichi di lavoro che richiedono interruzioni, la migrazione in tempo reale e l'aggiornamento in tempo reale. Unità: percentuale
NetworkIn	Metrica EC2. Amazon GameLift Servers Questa metrica rappresenta le prestazioni hardware di tutte le istanze attive in una flotta di sedi. Il numero di byte ricevuti dall'istanza su tutte le interfacce di rete. Identifica il volume del traffico di rete in entrata su un'applicazione di una singola istanza. Unità: byte
NetworkOut	Metrica EC2. Amazon GameLift Servers Questa metrica rappresenta le prestazioni hardware di tutte le istanze attive in una flotta di sedi. Il numero di byte inviati dall'istanza su tutte le interfacce di rete. Identifica il volume del traffico di rete in uscita verso un'applicazione di una singola istanza. Unità: byte

Metrica	Description
DiskReadBytes	Metrica EC2. Amazon GameLift ServersQuesta metrica rappresenta le prestazioni hardware di tutte le istanze attive in una flotta di sedi. Byte letti da tutti i volumi di instance store disponibili per l'istanza . Questo parametro viene utilizzato per determinare il volume dei dati che l'applicazione legge dal disco rigido dell'istanza. È possibile utilizzarlo per determinare la velocità dell'applicazione. Unità: byte
DiskWriteBytes	Metrica EC2. Amazon GameLift ServersQuesta metrica rappresenta le prestazioni hardware di tutte le istanze attive in una flotta di sedi. Byte scritti in tutti i volumi di instance store disponibili per l'istanza . Questo parametro viene utilizzato per determinare il volume dei dati che l'applicazione scrive sul disco rigido dell'istanza. È possibile utilizzarlo per determinare la velocità dell'applicazione. Unità: byte
DiskReadOps	Metrica EC2. Amazon GameLift ServersQuesta metrica rappresenta le prestazioni hardware di tutte le istanze attive in una flotta di sedi. Operazioni di lettura completate da tutti i volumi di instance store disponibili per l'istanza in un determinato periodo di tempo. Per calcolare le I/O operazioni medie al secondo (IOPS) per il periodo, dividi il totale delle operazioni nel periodo per il numero di secondi in quel periodo. Unità: numero

Metrica	Description
DiskWriteOps	Metrica EC2. Amazon GameLift Servers Questa metrica rappresenta le prestazioni hardware di tutte le istanze attive in una flotta di sedi. Operazioni di scrittura completate in tutti i volumi di instance store disponibili per l'istanza in un determinato periodo di tempo. Per calcolare le I/O operazioni medie al secondo (IOPS) per il periodo, dividi il totale delle operazioni nel periodo per il numero di secondi in quel periodo. Unità: numero

Processi server

Metrica	Description
ActiveServerProcesses	Processi server con stato ACTIVE (Attivo), ovvero processi in esecuzione e in grado di ospitare sessioni di gioco. Il numero include i processi server inattivi e quelli che ospitano una o più sessioni di gioco. Questo parametro misura la capacità di processi server totale corrente. Unità: numero CloudWatch Statistiche pertinenti: media, minima, massima Dimensioni: Ubicazione
HealthyServerProcesses	Processi server attivi che risultano integri. Questo è un parametro utile per tenere traccia dello stato complessivo dei server di gioco del parco istanze. Unità: numero

Metrica	Description
	CloudWatch Statistiche pertinenti: media, minima, massima Dimensioni: Ubicazione
PercentHealthyServerProcesses	Percentuale di tutti i processi server attivi che risultano integri (calcolata come $\text{HealthyServerProcesses} / \text{ActiveServerProcesses}$). Unità: percentuale CloudWatch Statistiche pertinenti: media, minima, massima Dimensioni: Ubicazione
ServerProcessAbnormalTerminations	Processi server che sono stati chiusi a causa di circostanze anomale dall'ultimo rapporto. Questo parametro include le terminazioni avviate dal servizio Amazon GameLift Servers. Ciò si verifica quando un processo del server smette di rispondere, segnala costantemente i controlli di integrità non riusciti o non termina in modo corretto (chiamando () ProcessEnding). Unità: numero CloudWatch Statistiche pertinenti: somma, media, minimo, massimo Dimensioni: Ubicazione

Metrica	Description
ServerProcessActivations	Processi server che sono passati correttamente dallo stato ACTIVATING (Attivazione in corso) allo stato ACTIVE (Attivo) dall'ultimo rapporto. I processi server non possono ospitare sessioni di gioco se non sono attivi. Unità: numero CloudWatch Statistiche pertinenti: somma, media, minima, massima Dimensioni: Ubicazione
ServerProcessTerminations	Processi server che sono stati chiusi dall'ultimo rapporto. Ciò include tutti i processi server che sono passati allo stato TERMINATED (Chiuso) per qualunque motivo, anche le terminazioni normali e non normali. Unità: numero CloudWatch Statistiche pertinenti: somma, media, minima, massima Dimensioni: Ubicazione

Sessioni di gioco

Metrica	Description
ActivatingGameSessions	Sessioni di gioco con stato ACTIVATING (Attivazione in corso), ovvero sessioni in fase di avvio. Le sessioni di gioco possono ospitare i giocatori se non sono attive. Numeri alti per un lungo periodo di tempo possono indicare che le sessioni di gioco non passano dallo stato ACTIVATING (Attivazione

Metrica	Description
	in corso) allo stato ACTIVE (Attivo). e può essere utilizzato con la scalabilità automatica. Unità: numero CloudWatch Statistiche pertinenti: media, minima, massima Dimensioni: Ubicazione
ActiveGameSessions	Sessioni di gioco con stato ACTIVE (Attivo), ovvero sessioni in grado di ospitare giocatori e che ospitano zero o più giocatori. Questo parametro misura il numero totale di sessioni di gioco attualmente ospitate. e può essere utilizzato con la scalabilità automatica. Unità: numero CloudWatch Statistiche pertinenti: media, minima, massima Dimensioni: Ubicazione

Metrica	Description
AvailableGameSessions	Processi server attivi e integri che non vengono attualmente utilizzati per ospitare una sessione di gioco e che possono avviare una nuova sessione di gioco senza ritardi per avviare nuovi processi o istanze del server. e può essere utilizzato con la scalabilità automatica.  Note Per le flotte che limitano le attivazioni simultanee di sessioni di gioco, utilizza la metrica. <code>ConcurrentAvailableGameSessions</code> Questa metrica rappresenta in modo più accurato il numero di nuove sessioni di gioco che possono iniziare senza alcun tipo di ritardo. Unità: numero CloudWatch Statistiche pertinenti: media, minima, massima Dimensioni: Ubicazione

Metrica	Description
ConcurrentActivatableGameSessions	Processi server attivi e integri che non vengono attualmente utilizzati per ospitare una sessione di gioco e che possono iniziare immediatamente una nuova sessione di gioco. Questa metrica differisce <code>AvailableGameSessions</code> nel modo seguente: non conta i processi del server che attualmente non possono attivare una nuova sessione di gioco a causa dei limiti alle attivazioni delle sessioni di gioco. (Vedi l'impostazione RuntimeConfiguration opzionale della flotta). <code>MaxConcurrentGameSessionActivations</code> Per le flotte che non limitano le attivazioni delle sessioni di gioco, questa metrica è identica a <code>AvailableGameSessions</code> Unità: numero CloudWatch Statistiche pertinenti: media, minima, massima Dimensioni: Ubicazione
PercentAvailableGameSessions	Percentuale delle slot di sessioni di gioco su tutti i processi server attivi (integri o non integri) che non sono attualmente in uso (calcolata come $\text{AvailableGameSessions} / [\text{ActiveGameSessions} + \text{AvailableGameSessions} + \text{unhealthy server processes}]$). e può essere utilizzato con la scalabilità automatica. Unità: percentuale CloudWatch Statistiche pertinenti: media Dimensioni: ubicazione

Metrica	Description
GameSessionInterruptions	Il numero di sessioni di gioco su istanze Spot che sono state interrotte. Unità: numero CloudWatch Statistiche pertinenti: somma, media, minima, massima Dimensioni: Ubicazione

Sessioni per giocatori

Metrica	Description
CurrentPlayerSessions	Le sessioni giocatore con stato ATTIVO (il giocatore è connesso a una sessione di gioco attiva) o stato RISERVATO (al giocatore è stato assegnato uno slot in una sessione di gioco ma non è ancora stato connesso). e può essere utilizzato con la scalabilità automatica. Unità: numero CloudWatch Statistiche pertinenti: media, minima, massima
PlayerSessionActivations	Le sessioni giocatore che passano dallo stato RISERVATO a quello ATTIVO dall'ultimo rapporto. Questo succede quando un giocatore si connette correttamente a una sessione di gioco attiva. Unità: numero CloudWatch Statistiche pertinenti: somma, media, minima, massima

Metriche della flotta di container

Le metriche della flotta di container forniscono dati di monitoraggio per l'hosting di giochi basato su container, tra cui lo stato del gruppo di container, l'utilizzo delle risorse e le metriche delle prestazioni.

Note

Quando si utilizza la soluzione per i parametri di telemetria, sono disponibili dashboard avanzati per il monitoraggio della flotta di container tramite [Amazon Managed Grafana](#). [the section called “Monitoraggio con metriche di telemetria del server”](#) Per le istruzioni di configurazione, consulta la pagina.

Metrica	Description
ActiveGameServerContainerGroups	Gruppi di contenitori attualmente attivi e pronti per ospitare sessioni di gioco. Unità: numero CloudWatch Statistiche pertinenti: media, minima, massima Dimensioni: Ubicazione
IdleGameServerContainerGroups	Gruppi di contenitori attivi ma che attualmente non ospitano sessioni di gioco. Unità: numero CloudWatch Statistiche pertinenti: media, minima, massima Dimensioni: ubicazione
PendingGameServerContainerGroups	Gruppi di contenitori in fase di avvio. Unità: numero

Metrica	Description
	CloudWatch Statistiche pertinenti: media, minima, massima Dimensioni: ubicazione
TerminatingGameServerContainerGroups	Gruppi di contenitori in fase di chiusura. Unità: numero CloudWatch Statistiche pertinenti: media, minima, massima Dimensioni: ubicazione
UnhealthyGameServerContainerGroupsReplaced	Gruppi di contenitori che sono stati sostituiti a causa di errori nei controlli di integrità. Unità: numero CloudWatch Statistiche pertinenti: somma, media, minimo, massimo Dimensioni: Ubicazione
ContainerCPUUtilizationPerInstance	Utilizzo medio della CPU in tutti i contenitori di un'istanza. Unità: percentuale CloudWatch Statistiche pertinenti: media, minima, massima Dimensioni: ubicazione

Metrica	Description
ContainerCPUReservation	Percentuale di risorse CPU riservate dai contenitori. Unità: percentuale CloudWatch Statistiche pertinenti: media, minima, massima Dimensioni: ubicazione
ContainerMemoryReservation	Percentuale di risorse di memoria riservate dai contenitori. Unità: percentuale CloudWatch Statistiche pertinenti: media, minima, massima Dimensioni: ubicazione
ContainerMemoryUtilization	Percentuale di memoria riservata utilizzata dai contenitori. Unità: percentuale CloudWatch Statistiche pertinenti: media, minima, massima Dimensioni: ubicazione
ContainerNetworkIn	Byte di rete ricevuti dai contenitori. Unità: byte CloudWatch Statistiche pertinenti: somma, media, minimo, massimo Dimensioni: Ubicazione

Metrica	Description
ContainerNetworkOut	Byte di rete inviati dai contenitori. Unità: byte CloudWatch Statistiche pertinenti: somma, media, minimo, massimo Dimensioni: Ubicazione
ContainerStorageReadBytes	Byte letti dallo storage dai contenitori. Unità: byte CloudWatch Statistiche pertinenti: somma, media, minimo, massimo Dimensioni: Ubicazione
ContainerStorageWriteBytes	Byte scritti in memoria dai contenitori. Unità: byte CloudWatch Statistiche pertinenti: somma, media, minima, massima Dimensioni: Ubicazione

DDoMetriche di protezione S (player gateway)

Amazon GameLift Servers le metriche del player gateway tengono traccia del traffico UDP che scorre attraverso il livello di protezione DDo S quando `PlayerGatewayMode` è impostato su o. `ENABLED` `REQUIRED` Queste metriche consentono di monitorare i modelli di traffico, identificare potenziali attacchi DDo S e monitorare le prestazioni dei relay.

Metrica	Description
PlayerGatewayPacketsIn	Numero di pacchetti UDP ricevuti dai giocatori e inoltrati ai server di gioco tramite il livello di protezione S. DDo Unità: numero CloudWatch Statistiche pertinenti: somma, media, minima, massima Dimensioni: FleetId, Ubicazione
PlayerGatewayBytesIn	Numero di byte ricevuti dai giocatori e inoltrati ai server di gioco tramite il livello di protezione DDo S. Unità: byte CloudWatch Statistiche pertinenti: somma, media, minimo, massimo Dimensioni: FleetId, Ubicazione
PlayerGatewayPacketsOut	Numero di pacchetti UDP ricevuti dai server di gioco e inoltrati ai giocatori tramite il DDo livello di protezione S. Unità: numero CloudWatch Statistiche pertinenti: somma, media, minima, massima Dimensioni: FleetId, Ubicazione
PlayerGatewayBytesOut	Numero di byte ricevuti dai server di gioco e inoltrati ai giocatori tramite il livello di protezione DDo S. Unità: byte CloudWatch Statistiche pertinenti: somma, media, minima, massima

Metrica	Description
	Dimensioni: FleetId, Ubicazione
PlayerGatewayPacketsThrottled	Numero di pacchetti in entrata eliminati a causa della limitazione della velocità da parte del livello di protezione DDo S. Unità: numero CloudWatch Statistiche pertinenti: somma, media, minimo, massimo Dimensioni: FleetId, Ubicazione
PlayerGatewayBytesThrottled	Il numero di byte in entrata è diminuito a causa della limitazione della velocità da parte del livello di protezione DDo S. Unità: byte CloudWatch Statistiche pertinenti: somma, media, minimo, massimo Dimensioni: FleetId, Ubicazione
PlayerGatewayPlayerSessions	Sessioni di gioco attualmente DDo protette dal livello di protezione S. Unità: numero CloudWatch Statistiche pertinenti: media, minima, massima Dimensioni: FleetId, Ubicazione

Parametri Amazon GameLift Servers per code

Il namespace `Amazon GameLift` include i seguenti parametri relativi all'attività in una coda di posizionamento delle sessioni di gioco. Le code vengono utilizzate con una soluzione Amazon

GameLift Servers gestita. Il Amazon GameLift Servers servizio invia metriche a CloudWatch ogni minuto.

Metrica	Description
AverageWaitTime	Il tempo medio di attesa prima che le richieste di posizionamento delle sessioni di gioco in coda con stato IN SOSPESO vengano soddisfatte. Unità: secondi CloudWatch Statistiche pertinenti: media, minima, massima, somma Dimensioni: ubicazione
FirstChoiceNotViable	Sessioni di gioco correttamente piazzate ma NON nel parco istanze di prima scelta, perché tale parco istanze non era considerato realizzabile (come un parco istanze Spot con un alto tasso di interruzione). Questa metrica si basa sul costo, non sulla latenza. La flotta di prima scelta è la prima flotta elencata in coda oppure, quando una richiesta di posizionamento include dati sulla latenza dei giocatori, è la prima flotta scelta in base alla priorità. FleetIQ Se non ci sono parchi Spot realizzabili, potrebbe essere selezionato qualsiasi parco istanze in quella regione. Unità: numero Statistiche pertinenti: media, minima, massima, somma CloudWatch
FirstChoiceOutOfCapacity	Sessioni di gioco correttamente piazzate ma NON nel parco istanze di prima scelta, perché tale parco istanze non aveva risorse disponibili. La flotta di prima scelta è la prima flotta elencata in coda oppure, quando una richiesta di posizionamento

Metrica	Description
	include dati sulla latenza dei giocatori, è la prima flotta scelta in base alla priorità definita. FleetIQ Unità: numero CloudWatch Statistiche pertinenti: media, minima, massima, somma
LowestLatencyPlacement	Le sessioni di gioco correttamente piazzate in una regione che offre la più bassa latenza possibile della coda per i giocatori. Questo parametro viene emesso solo quando i dati di latenza del giocatore sono inclusi nella richiesta di posizionamento. Unità: numero CloudWatch Statistiche pertinenti: media, minima, massima, somma
LowestPricePlacement	Sessioni di gioco inserite con successo in una flotta al prezzo più basso possibile della coda per la regione selezionata. Il parco istanze può essere un parco istanze Spot o un'istanza on demand se la coda non presenta parchi istanze Spot. Questo parametro viene emesso solo quando i dati di latenza del giocatore sono inclusi nella richiesta di posizionamento. Unità: numero CloudWatch Statistiche pertinenti: media, minima, massima, somma

Metrica	Description
Placement <region name>	Le sessioni di gioco correttamente piazzate nei parchi istanze posizionati nella regione specificata. Questo parametro scompone il parametro PlacementsSucceeded per regione. Unità: numero CloudWatch Statistiche pertinenti: somma
PlacementsCanceled	Le richieste di posizionamento delle sessioni di gioco che sono state annullate prima del timeout dall'ultimo rapporto. Unità: numero CloudWatch Statistiche pertinenti: media, minima, massima, somma
PlacementsFailed	Le richieste di posizionamento delle sessioni di gioco non andate a buon fine per qualsiasi motivo dall'ultimo rapporto. Unità: numero CloudWatch Statistiche pertinenti: media, minima, massima, somma
PlacementsStarted	Le nuove richieste di posizionamento delle sessioni di gioco aggiunte alla coda dall'ultimo rapporto. Unità: numero CloudWatch Statistiche pertinenti: media, minima, massima, somma

Metrica	Description
PlacementsSucceeded	Le richieste di posizionamento delle sessioni di gioco che hanno avuto come esito una nuova sessione di gioco dall'ultimo rapporto. Unità: numero CloudWatch Statistiche pertinenti: media, minima, massima, somma
PlacementsTimedOut	Le richieste di posizionamento delle sessioni di gioco che hanno raggiunto il limite di timeout della coda senza venire soddisfatte dall'ultimo rapporto. Unità: numero CloudWatch Statistiche pertinenti: media, minima, massima, somma
QueueDepth	Il numero di richieste di posizionamento delle sessioni di gioco in coda con stato IN SOSPESO. Unità: numero CloudWatch Statistiche pertinenti: media, minima, massima, somma Dimensioni: ubicazione

Parametri Amazon GameLift Servers per l'abbinamento

Lo spazio dei nomi Amazon GameLift include i parametri dell'attività FlexMatch per le configurazioni di abbinamento e le regole di abbinamento. L'abbinamento FlexMatch viene utilizzato con una soluzione Amazon GameLift Servers gestita. Il Amazon GameLift Servers servizio invia metriche a CloudWatch ogni minuto.

Per ulteriori informazioni sulla sequenza delle attività di matchmaking, vedi [Come funziona Amazon GameLift ServersFlexMatch](#).

Configurazioni di matchmaking

Metrica	Description
CurrentTickets	Le richieste di abbinamento attualmente in fase di elaborazione o in attesa di essere elaborate. Unità: numero CloudWatch Statistiche pertinenti: Media, Minimo, Massimo, Somma
MatchAcceptancesTimedOut	Per le configurazioni di abbinamento che richiedono o accettazione, i potenziali abbinamenti scaduti durante l'accettazione dall'ultimo rapporto. Unità: numero CloudWatch Statistiche pertinenti: somma
MatchesAccepted	Per le configurazioni di abbinamento che richiedono accettazione, i potenziali abbinamenti che sono stati accettati dall'ultimo rapporto. Unità: numero CloudWatch Statistiche pertinenti: Sum
MatchesCreated	I potenziali abbinamenti che sono stati creati dall'ultimo rapporto. Unità: numero CloudWatch Statistiche pertinenti: Sum
MatchesPlaced	Gli abbinamenti che sono stati posizionati correttamente in una sessione di gioco dall'ultimo rapporto. Unità: numero CloudWatch Statistiche pertinenti: Sum

Metrica	Description
MatchesRejected	Per le configurazioni di abbinamento che richiedono accettazione, i potenziali abbinamenti che sono stati rifiutati da almeno un giocatore dall'ultimo rapporto. Unità: numero CloudWatch Statistiche pertinenti: Sum
PlayersStarted	I giocatori in ticket di abbinamento che sono stati aggiunti dall'ultimo rapporto. Unità: numero CloudWatch Statistiche pertinenti: Sum
TicketsFailed	Le richieste di abbinamento che hanno prodotto un errore dall'ultimo rapporto. Unità: numero CloudWatch Statistiche pertinenti: Sum
TicketsStarted	Le nuove richieste di abbinamento che sono state create dall'ultimo rapporto. Unità: numero CloudWatch Statistiche pertinenti: Sum
TicketsTimedOut	Le richieste di abbinamento che hanno raggiunto il limite di timeout dall'ultimo rapporto. Unità: numero CloudWatch Statistiche pertinenti: Sum

Metrica	Description
TimeToMatch	Per le richieste di abbinamento che vengono inserite in un abbinamento potenziale prima dell'ultimo rapporto, il tempo che intercorre tra la creazione del ticket e la creazione di abbinamenti potenziali. Unità: secondi CloudWatch Statistiche pertinenti: campioni di dati, media, minima, massima
TimeToTicketCancel	Per le richieste di abbinamento che sono state annullate prima dell'ultimo rapporto, il tempo che intercorre tra la creazione e l'annullamento del ticket. Unità: secondi CloudWatch Statistiche pertinenti: campioni di dati, media, minima, massima
TimeToTicketSuccess	Per le richieste di abbinamento che sono state completate prima dell'ultimo rapporto, il tempo che intercorre tra la creazione del ticket e il corretto posizionamento dell'abbinamento. Unità: secondi CloudWatch Statistiche pertinenti: campioni di dati, media, minima, massima

Regole di abbinamento

Metrica	Description
RuleEvaluationsPassed	Le valutazioni delle regole durante il processo di abbinamento superato dall'ultimo rapporto. Questo parametro si limita alle prime 50 regole.

Metrica	Description
	Unità: numero
	CloudWatchStatistiche pertinenti: somma
RuleEvaluationsFailed	Le valutazioni delle regole durante il processo di abbinamento non superato dall'ultimo rapporto. Questo parametro si limita alle prime 50 regole.
	Unità: numero
	CloudWatch Statistiche pertinenti: Sum

Parametri Amazon GameLift Servers per FleetIQ

Lo spazio dei nomi Amazon GameLift include i parametri per il gruppo di server di gioco FleetIQ e l'attività del server di gioco come parte di una soluzione standalone FleetIQ per l'hosting di giochi. Il Amazon GameLift Servers servizio invia metriche a CloudWatch ogni minuto. Consulta anche [Monitoraggio dei gruppi e delle istanze di Auto Scaling con Amazon CloudWatch nella Guida per l'utente](#) di Amazon EC2 Auto Scaling.

Metrica	Description
AvailableGameServers	Server di gioco disponibili per eseguire un'esecuzione di un gioco non attualmente occupati dal gameplay. Questo numero include i server di gioco che sono stati registrati ma sono ancora in stato DISPONIBILE.
	Unità: numero
	CloudWatch Statistiche Amazon pertinenti: Sum
	Dimensioni: GameServerGroup
UtilizedGameServers	Server di gioco che sono attualmente occupati con gameplay. Questo numero include i server di gioco che sono nello stato UTILIZED (UTILIZZATO).

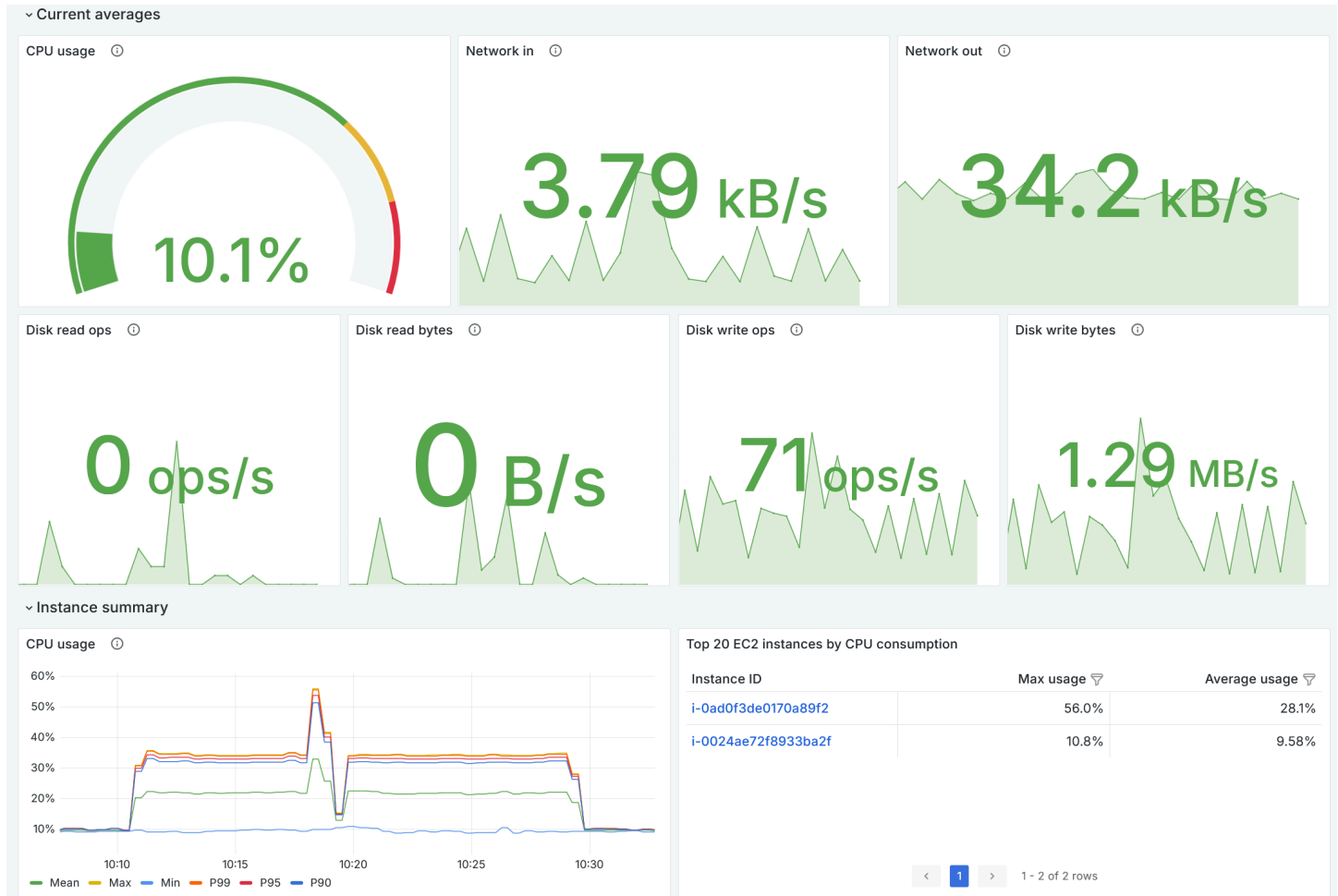
Metrica	Description
	Unità: numero CloudWatch Statistiche Amazon pertinenti: Sum Dimensioni: GameServerGroup
DrainingAvailableGameServers	Server di gioco su istanze programmate per la terminazione che attualmente non supportano il gameplay. Questi server di gioco sono la priorità più bassa da rivendicare in risposta a una nuova richiesta di registrazione. Unità: numero CloudWatch Statistiche Amazon pertinenti: Sum Dimensioni: GameServerGroup
DrainingUtilizedGameServers	Server di gioco su istanze programmate per la terminazione che attualmente supportano la modalità di gioco. Unità: numero CloudWatch Statistiche Amazon pertinenti: Sum Dimensioni: GameServerGroup

Metrica	Description
PercentUtilizedGameServers	Porzione di server di gioco che attualmente supportano le esecuzioni di gioco. Questa metrica indica la quantità di capacità del server di gioco attualmente in uso. È utile per guidare una politica di ridimensionamento automatico che può aggiungere e rimuovere dinamicamente istanze in base alla domanda dei giocatori. Unità: percentuale CloudWatch Statistiche Amazon pertinenti: media, minima, massima Dimensioni: GameServerGroup
GameServerInterruptions	Server di gioco su istanze Spot che sono state interrotte a causa della disponibilità Spot limitata. Unità: numero CloudWatch Statistiche Amazon pertinenti: Sum Dimensioni: GameServerGroup, InstanceType
InstanceInterruptions	Istanze Spot che sono state interrotte a causa della disponibilità limitata. Unità: numero CloudWatch Statistiche Amazon pertinenti: Sum Dimensioni: GameServerGroup, InstanceType

Monitoraggio con metriche di telemetria del server

Amazon GameLift Servers può essere configurato per raccogliere e pubblicare metriche di telemetria per i server di gioco in esecuzione su flotte gestite di Amazon EC2 e Container. Queste metriche

diventano disponibili dopo aver implementato il raccogliatore di telemetria con la build del server. Il sistema di metriche supporta tutti SDKs (C++, C#, Go), tutti i plugin (Unreal, Unity) e il Game Server Wrapper. Amazon GameLift Servers I dati delle metriche vengono trasferiti ad [Amazon Managed Service for the section called “Monitora con CloudWatch” Prometheus](#) e alle dashboard di [Amazon Managed Grafana](#) (consigliate per la visualizzazione).



Monitora le prestazioni dei tuoi server di gioco utilizzando dashboard preconfigurate in Amazon Managed Grafana o Amazon CloudWatch

Vantaggi delle metriche di telemetria

Il sistema di metriche di telemetria offre cinque vantaggi chiave:

- Metriche specifiche del motore di gioco: i plug-in del motore di gioco (Unreal, Unity) forniscono un'integrazione nativa con metriche prestazionali specifiche del motore, come il tick time del server, il frame rate e l'utilizzo delle risorse a livello di motore, fondamentali per l'ottimizzazione delle prestazioni di gioco.

- Supporto per metriche personalizzate: definisci e monitora le tue metriche specifiche del gioco utilizzando le chiamate alle funzioni SDK del server per monitorare eventi di gioco personalizzati, prestazioni di business logic e punti dati specifici dell'applicazione importanti per il tuo gioco.
- Raccolta automatizzata: le metriche fluiscono automaticamente dopo l'implementazione del raccoglitore di telemetria senza bisogno di strumentazione aggiuntiva e integrazione diretta con Amazon Managed [Service](#) per Prometheus e Amazon. CloudWatch
- Monitoraggio a più livelli: metriche a livello di flotta per capacità e scalabilità, metriche a livello di istanza per l'utilizzo delle risorse e metriche delle sessioni di gioco per il monitoraggio delle prestazioni.
- Compatibilità universale: funziona con tutti gli ambienti di sviluppo Amazon GameLift Servers supportati, è integrato con tutti i server e supporta nativamente i plugin dei motori di gioco. SDKs

Note

Le metriche di telemetria sono disponibili per le flotte gestite di Amazon GameLift Servers Amazon EC2 o container che eseguono Amazon Linux 2023 o Windows.

Prima di iniziare

Risorse richieste AWS

- AWS account configurato per Amazon GameLift Servers.
- Flotta gestita in esecuzione su:
 - Amazon EC2 con sistemi operativi supportati, OPPURE
 - Contenitori con Amazon Linux 2023
- Autorizzazioni IAM appropriate

Requisiti IAM

Le seguenti autorizzazioni IAM sono richieste solo se prevedi di utilizzare il servizio corrispondente:

- [Amazon Managed Service for Prometheus](#) (richiesto solo se si pubblicano metriche su Prometheus)
 - Autorizzazione `aps:RemoteWrite`

- Accesso al tuo spazio di lavoro Prometheus
- Amazon CloudWatch (richiesto solo se si pubblicano metriche su Amazon CloudWatch)
 - Autorizzazione `cloudwatch:PutMetricData`
 - Accesso ai namespace delle metriche
- [Amazon Managed Grafana](#) (richiesto solo se si utilizzano le dashboard Grafana)
 - Autorizzazione `grafana:Read`
 - Configurazione SSO per l'accesso alla dashboard

Implementazione

Seleziona il percorso di implementazione in base al tuo ambiente di sviluppo:

Implementazione SDK

Tipo SDK	Configurazione SDK	Parametri personalizzati	Documentazione di riferimento delle API
Vai a SDK	Guida completa alla configurazione	API Go Metrics	Azioni e tipi di dati Go
SDK C#	Guida completa alla configurazione	API C# Metrics	Azioni e tipi di dati C#
SDK C++	Guida completa alla configurazione	API C++ Metrics	Azioni e tipi di dati C++

implementazione del plugin

Plug-in	Configurazione del plugin	Parametri personalizzati	Documentazione di riferimento delle API
Irreale	Guida completa alla configurazione	API Unreal Metrics	Azioni e tipi di dati irreali

Plug-in	Configurazione del plugin	Parametri personalizzati	Documentazione di riferimento delle API
Unità	Guida alla configurazione completa	API Unity Metrics	Azioni e tipi di dati C#

Workflow di implementazione

Ogni implementazione segue un processo in due fasi:

1. Guida completa all'installazione (Metrics.md): implementazione dell'infrastruttura, configurazione dell' AWS infrastruttura, configurazione della flotta e configurazione del dashboard Grafana.
2. Guida all'implementazione dell'API (Custom_metrics.md): utilizzo dell'SDK specifico per lingua, tipi di metriche, creazione di metriche personalizzate e configurazione avanzata.

Verifica

1. Convalida il flusso dei parametri controllando l'[area di lavoro di Amazon Managed Service for Prometheus](#) o la console Amazon per verificare la presenza di dati di telemetria in entrata. CloudWatch
2. Verifica la visibilità del pannello di controllo in [Amazon Managed Grafana](#) utilizzando i dashboard predefiniti.
3. Testa le metriche personalizzate verificando che compaiano nelle dashboard di monitoraggio.

Note

Dopo aver completato l'implementazione, torna a questa pagina e vai alla pagina [Metriche disponibili](#).

Metriche disponibili

Le metriche si dividono in tre categorie:

- Metriche raccolte automaticamente
- Metriche fornite da SDK

- parametri personalizzati

Raccolta automatica di metriche

Non sono necessarie modifiche al codice per queste metriche:

Parametri dell'istanza

Tipo di metrica	Description	Caso d'uso
CPU	Percentuale di utilizzo per istanza	Monitoraggio delle risorse
Memoria	Utilizzo e percentuale della memoria fisica	Pianificazione della capacità
I/O di rete	Byte e pacchetti inviati/ricevuti	Stato della connessione
I/O del disco	Operazioni e velocità effettiva di lettura/s crittura	Prestazioni di archiviazione

Metriche della flotta

Tipo di metrica	Description	Caso d'uso
Istanze attive	Conteggio delle istanze in esecuzione	Scalabilità del parco veicoli
Sessioni di gioco	Sessioni attive e disponibili	Gestione della capacità
Sessioni di gioco interrotte	Sessioni di gioco che si sono bloccate	Monitoraggio degli errori

Metriche fornite dall'SDK

Richiede chiamate alle funzioni SDK nel codice:

Metriche di temporizzazione del server

Metrica	Description	Implementazione
Delta Time del server	Differenza di tempo tra il segno di spunta corrente del server e il segno di spunta del server precedente. Misura la coerenza del tick rate del server	Chiama <code>GetDeltaTime ()</code>
Tick Rate del server	Mostra il numero di volte al secondo in cui il server elabora gli aggiornamenti	Calcolato automaticamente
Orario di selezione del server	Il tempo impiegato dal server per elaborare un singolo segno di spunta o aggiornamento	Chiama <code>GetTickTime ()</code>
Server World Tick Time	Il tempo impiegato dal server per aggiornare il mondo di gioco con ogni segno di spunta	Chiama <code>GetWorldUpdateTime ()</code>

Implementazione: per chi è indipendente dal motore SDKs (C++, C#, Go), implementa queste metriche richiamando le funzioni SDK dal loop di gioco con valori di temporizzazione calcolati. Per i plugin del motore (Unreal, Unity), queste metriche vengono acquisite automaticamente tramite l'integrazione del motore.

Metriche di rete

Metrica	Description	Implementazione
Connessioni	Il numero totale di connessioni di rete stabilite dal server	Automatico dopo <code>InitMetrics ()</code>
Rete I/O (byte)	Il numero totale di byte inviati e ricevuti dal server sulla rete	Automatico dopo <code>InitMetrics ()</code>
Rete I/O (pacchetti)	Il numero totale di pacchetti di rete inviati e ricevuti dal server	Automatico dopo <code>InitMetrics ()</code>

Metrica	Description	Implementazione
Perdita di pacchetti	La percentuale di pacchetti di rete che vengono persi durante la trasmissione	Automatico dopo InitMetrics ()

Implementazione: integra le chiamate alle funzioni SDK con la tua libreria di rete. L'SDK fornisce indicazioni per diverse implementazioni di rete.

Parametri di elaborazione

Metrica	Description	Implementazione
Utilizzo della CPU (%)	La percentuale di risorse della CPU utilizzate dal processo del server di gioco	Automatico dopo InitMetrics ()
Utilizzo della memoria (unità)	La quantità totale di memoria consumata dai processi del server	Automatico dopo InitMetrics ()
Utilizzo della memoria fisica (%)	La percentuale della memoria fisica totale del server attualmente utilizzata	Automatico dopo InitMetrics ()
Stato del server	Stato di integrità del server di gioco	Automatico dopo InitMetrics ()

Implementazione: queste metriche vengono raccolte automaticamente dall'SDK per ogni processo di sessione di gioco.

Organizzazione del pannello di controllo per processo

Le metriche per processo sono disponibili in due dashboard specializzati:

- Dashboard delle prestazioni del server: contiene gli orari del server (delta time, tick rate, tick time, world tick time), le metriche di rete (connessioni, I/O byte/pacchetti, perdita di pacchetti), l'utilizzo della memoria e l'utilizzo della CPU per le singole sessioni di gioco.
- Dashboard delle prestazioni dell'istanza: include le tabelle «Le migliori N sessioni di gioco che consumano la memoria» e le «N sessioni di gioco che consumano più CPU» che aiutano a identificare i processi che contribuiscono maggiormente al consumo di risorse dell'istanza. Facendo clic sui collegamenti alle sessioni di gioco è possibile eseguire un'analisi più approfondita delle metriche dettagliate.

Casi d'uso delle metriche per processo

Le per-game-session metriche per processo/ supportano i seguenti scenari di monitoraggio:

- **Analisi approfondita delle prestazioni:** quando le prestazioni di a sono peggiorate a causa di processi o sessioni di gioco specifici, le metriche per processo aiutano a identificare quale processo host/instance ha causato il problema nelle tabelle delle sessioni di gioco che consumano più CPU e memoria.
- **Indagine sugli arresti anomali di un server di gioco:** quando una sessione di gioco si blocca, queste metriche aiutano a determinare se l'arresto è dovuto a esaurimento della memoria, sovraccarico della CPU o problemi di larghezza di banda della rete.
- **Analizza i problemi segnalati dai giocatori:** quando i giocatori segnalano ritardi o interruzioni durante il gioco, le metriche basate sui singoli processi aiutano a identificare i punti deboli nella CPU, nella memoria, nella rete, nel tempo di tick o nel tempo di aggiornamento globale.
- **Identifica le variazioni delle prestazioni nelle diverse build:** le metriche relative al tempo di tick, alla frequenza di tick e al tempo di aggiornamento mondiale consentono agli sviluppatori di misurare come cambiano le prestazioni di gioco su diverse build di server.
- **Rileva ritardi e rallentamenti nel gameplay:** le metriche relative al tempo di tick, alla frequenza di tick e al tempo di aggiornamento mondiale riflettono la velocità con cui il server aggiorna il gioco, con un impatto diretto sull'esperienza del cliente.
- **Analisi comparativa:** identifica in che modo i diversi scenari di gioco influiscono sulle prestazioni del server in base a fattori come il numero di giocatori, la modalità di gioco e altre variabili.

Organizzazione del pannello di controllo

Le metriche sono organizzate in dashboard specializzate in [Amazon Managed Grafana](#) per diversi scenari di monitoraggio. Le dashboard disponibili dipendono dal tipo di flotta:

EC2 Dashboard della flotta

- **EC2 Dashboard panoramica della flotta:** capacità della flotta di alto livello, informazioni sulla scalabilità, giocatori simultanei (CCU), istanze, capacità dei giocatori e sessioni di gioco interrotte.
- **Dashboard di panoramica delle istanze:** metriche aggregate a livello di host per tutte le istanze, tra cui l'utilizzo medio di CPU, memoria, rete e disco.
- **Dashboard delle prestazioni delle istanze:** metriche dettagliate per le singole istanze con le tabelle «Le migliori N sessioni di gioco che consumano memoria» e «Le N sessioni di gioco che consumano più CPU» per identificare i processi che richiedono molte risorse.

- Dashboard delle prestazioni del server (EC2): tempistica dei loop di gioco, prestazioni di rete, memoria e parametri della CPU per le singole sessioni di gioco su istanze. EC2

Dashboard Container Fleet

- Dashboard di panoramica sulla flotta di container: panoramica di alto livello sull'utilizzo delle risorse della flotta di container, inclusa la prenotazione della CPU, l'utilizzo della memoria e lo stato del gruppo di container.
- Dashboard delle prestazioni dei container: metriche dettagliate per i singoli contenitori nell'ambito di attività ECS specifiche, tra cui utilizzo della CPU, utilizzo della memoria, I/O di rete e prestazioni di archiviazione.
- Dashboard delle prestazioni del server (Container): tempistica dei loop di gioco, prestazioni di rete, memoria e parametri della CPU per le singole sessioni di gioco in contenitori.

Per informazioni dettagliate sulla dashboard e istruzioni per l'uso, consulta [the section called "Organizzazione e utilizzo del pannello di controllo"](#)

Come funziona

Il sistema di metriche di telemetria segue un semplice flusso di dati in quattro fasi dai server di gioco ai dashboard di visualizzazione.

Raccolta: il tuo server di gioco, integrato con l'SDK o il plug-in GameLift Server, invia automaticamente le metriche a un raccogliatore di telemetria locale in esecuzione sulla stessa istanza. L'SDK acquisisce sia metriche automatiche (ciclo di vita del server, utilizzo delle risorse) sia metriche personalizzate definite nel codice.

Elaborazione: il raccogliatore di telemetria aggrega le metriche del server di gioco e le combina con i dati sulle prestazioni a livello di istanza (CPU, memoria, rete, utilizzo del disco). Ciò fornisce un quadro completo delle prestazioni del gioco e dell'infrastruttura sottostante.

Archiviazione: le metriche elaborate vengono esportate nel magazzino di metriche di tua scelta: [Amazon Managed Service for Prometheus per lo storage di serie temporali ad alte prestazioni](#), [Amazon per l'integrazione dei servizi](#) o entrambi. CloudWatch AWS Tutte le trasmissioni di dati sono autenticate e crittografate.

Visualizzazione: [Amazon Managed Grafana](#) si collega al tuo magazzino di metriche per visualizzare dashboard predefinite. GameLift Queste dashboard forniscono panoramiche della flotta, dettagli sulle

prestazioni dei server e visualizzazioni di monitoraggio dei container che ti aiutano a monitorare e risolvere i problemi della tua infrastruttura di hosting dei giochi.

Note

Tutte le trasmissioni metriche tra il server di gioco e il raccogliore di telemetria avvengono localmente sull'istanza per motivi di sicurezza. Solo il raccogliore comunica con i servizi utilizzando l'autenticazione corretta. AWS

Organizzazione e utilizzo del pannello di controllo

Visualizza le tue metriche su dashboard complete in [Amazon Managed Grafana](#). Le dashboard disponibili dipendono dal tipo di flotta:

Disponibilità del pannello di controllo per tipo di flotta

La tabella seguente mostra quali dashboard sono disponibili per ogni tipo di flotta:

Pannello di controllo	Tipo di parco istanze	Description
EC2 Panoramica della flotta	EC2 Flotta	Visualizza informazioni sui giocatori simultanei (CCU), sulle istanze e sulla capacità dei giocatori
Panoramica delle istanze	EC2 Flotta	Visualizza l'utilizzo medio di CPU, memoria e rete in tutte le istanze del parco istanze
Prestazioni dell'istanza	EC2 Flotta	Visualizza metriche dettagliate (CPU, memoria, disco, rete) per una singola istanza
Panoramica della flotta di container	Flotta di container	Visualizza l'utilizzo medio delle risorse di tutti i container in una flotta di container gestita

Pannello di controllo	Tipo di parco istanze	Description
Prestazioni dei container	Flotta di container	Visualizza metriche dettagliate dei singoli contenitori all'interno di un'attività ECS specifica
Prestazioni del server	Entrambi	Visualizza le prestazioni di rete, memoria e runtime di uno specifico processo del server di gioco (versioni separate per flotte di container EC2 e flotte)

EC2 Flotte gestite:

- EC2 La panoramica della flotta fornisce informazioni di alto livello sulla capacità della flotta e sulla scalabilità.
- Utilizza le dashboard Instances Overview e Instance Performance per il monitoraggio a livello di host.
- Metriche raccolte tramite il ricevitore hostmetrics per la visibilità a livello di sistema.
- Concentrati sull'utilizzo e sulle prestazioni delle risorse dell' EC2 istanza.
- Server Performance (EC2) monitora le metriche delle applicazioni dei server di gioco indipendentemente dall'infrastruttura sottostante.

Flotte di container gestite:

- Utilizza i dashboard Container Fleet Overview e Container Performance per il monitoraggio delle attività ECS e a livello di container.
- Metriche raccolte tramite ECS Container Receiver per la visibilità dei carichi di lavoro containerizzati.
- Concentrati sull'aggregazione a livello di attività e sull'isolamento delle risorse dei container.
- Server Performance (Container) monitora le metriche delle applicazioni dei server di gioco indipendentemente dall'infrastruttura sottostante.

EC2 Dashboard di panoramica della flotta

Questa dashboard offre una panoramica di alto livello dell'utilizzo e della capacità della flotta a livello globale e per località. Presenta grafici che mostrano il conteggio degli arresti, degli avvii e degli arresti anomali dei server di gioco, nonché la percentuale di server di gioco sani. Puoi filtrare per FleetID e Location.

Metriche di panoramica della flotta

La tabella seguente mostra le metriche disponibili nella dashboard Fleet Overview:

Pannello di controllo	Nome parametro	Definizione
CCU e capacità globali	CCU	Mostra il numero di utenti simultanei in tutti i server di gioco in tutte le istanze in tutte le località
	CCU globale	Mostra il numero di utenti simultanei in tutti i server di gioco in tutte le istanze a livello globale
	Istanze attive	Mostra il numero totale di istanze attive nel parco istanze
	Processi attivi	Mostra il numero di processi attivi del server di gioco pronti per ospitare una sessione di gioco
	Il server di gioco si avvia	Mostra il numero di sessioni di gioco iniziate in tutta la flotta
CCU e capacità globali	Server di gioco sani	Mostra la percentuale media di server di gioco che risultano integri in Amazon GameLift Servers tutto il parco macchine
	Sessioni di gioco bloccate	Mostra la sessione IDs di gioco delle sessioni di gioco bloccate. Fai clic sul link dell'ARN della sessione di gioco per accedere alla dashboard Server Performance

Pannello di controllo	Nome parametro	Definizione
Ubicazione CCU	Location Players	Mostra il numero di giocatori simultanei in una posizione, inclusi tutti i server di gioco in tutte le istanze presenti nella posizione
	Capacità di localizzazione	Mostra l'utilizzo della capacità (%) in una località e la percentuale di server di gioco in uso nella posizione

Note

Le metriche CCU richiedono l'implementazione nel codice del server di gioco. Queste metriche non vengono raccolte automaticamente e devono essere implementate e riportate dall'applicazione.

Panoramica delle istanze (dashboard)

Questa dashboard fornisce metriche aggregate a livello di host per tutte le istanze del parco istanze. Le medie attuali mostrano lo stato generale delle istanze. Quando le prestazioni peggiorano, controlla l'utilizzo della CPU, il consumo di memoria, la rete e il consumo del disco per individuare eventuali colli di bottiglia. Puoi filtrare per FleetID e Location.

Panoramica delle istanze (metriche)

La tabella seguente mostra le metriche disponibili nella dashboard di panoramica delle istanze:

Pannello di controllo	Nome parametro	Definizione
Riassunto dell'istanza	Utilizzo medio della CPU	Il riepilogo delle istanze tiene traccia dell'utilizzo medio delle risorse per istanza. Percentuale media di risorse CPU utilizzate in tutte le istanze del parco istanze. I modelli di utilizzo elevato richiedono il controllo delle metriche delle singole istanze e della distribuzione delle risorse

Pannello di controllo	Nome parametro	Definizione
	Utilizzo massimo della CPU	Massimo utilizzo della CPU registrato tra tutte le istanze del parco istanze nel periodo di tempo
	Utilizzo della CPU per istanza	Utilizzo individuale della CPU per ogni istanza del parco, che consente l'identificazione degli hotspot di risorse
Riassunto dell'istanza	Utilizzo medio della memoria	Percentuale media di RAM utilizzata in tutte le istanze del parco istanze. I modelli di utilizzo elevato richiedono il controllo dei parametri delle singole istanze e della distribuzione delle risorse, ad esempio il consumo di memoria
	Utilizzo massimo della memoria	Massimo utilizzo della memoria registrato tra tutte le istanze del parco istanze nel periodo di tempo
	Utilizzo della memoria per istanza	Utilizzo individuale della memoria per ogni istanza del parco istanze, che aiuta a identificare i carichi di lavoro che richiedono molta memoria
Riepilogo della rete	Rete I/O (byte)	Il riepilogo della rete riflette la connettività media delle istanze. Volume medio del traffico di rete (inviato e ricevuto) su tutte le istanze del parco istanze. Per problemi di latenza o connettività, controlla le metriche delle singole istanze, come le frequenze di rete I/O e di pacchetti
	Rete I/O (pacchetti)	Frequenza media dei pacchetti di rete (inviati e ricevuti) su tutte le istanze del parco istanze
	Utilizzo della rete per istanza	Utilizzo della rete individuale per ogni istanza, utile per identificare i colli di bottiglia della rete

Pannello di controllo	Nome parametro	Definizione
Riepilogo del disco	I/O Operazioni su disco	Il riepilogo del disco indica le prestazioni medie del disco delle istanze. read/write Operazioni i medie su disco su tutte le istanze del parco istanze. I tempi di risposta lenti suggeriscono di esaminare le operazioni su disco, i tempi di I/O attesa e le operazioni in sospeso a livello di singola istanza
	Throughput del disco I/O	read/write Velocità effettiva media del disco su tutte le istanze del parco istanze
	Utilizzo del disco per istanza	Utilizzo individuale del disco per ogni istanza, che aiuta a identificare i problemi di prestazioni dello storage

Note

Le metriche a livello di istanza vengono raccolte tramite il ricevitore hostmetrics e forniscono visibilità a livello di sistema sulle prestazioni dell'infrastruttura del parco macchine. Utilizza questa dashboard per identificare le tendenze generali sullo stato della flotta e analizzare i singoli casi in cui vengono rilevati problemi di prestazioni.

Dashboard delle prestazioni delle ist

Questa dashboard fornisce metriche dettagliate sulle prestazioni per le singole istanze. Le medie attuali mostrano lo stato generale dell'istanza. Quando le prestazioni peggiorano, controlla l'utilizzo della CPU, della memoria e del file system per individuare eventuali colli di bottiglia. Presenta le tabelle «Top N Memory Consuming Game Sessions» e «Top N CPU Consuming Game Sessions» che aiutano a identificare i processi che contribuiscono maggiormente al consumo di risorse delle istanze. Facendo clic sui collegamenti alle sessioni di gioco è possibile eseguire un'analisi più approfondita delle metriche dettagliate. Puoi filtrare in base a un ID di istanza specifico.

Metriche delle prestazioni dell'istanza

La tabella seguente mostra le metriche disponibili nella dashboard delle prestazioni delle istanze:

Pannello di controllo	Nome parametro	Definizione
Riepilogo dei caricamenti	Le migliori sessioni di gioco che consumano N memoria	Il riepilogo del caricamento tiene traccia dell'utilizzo delle risorse da parte dei server di gioco. Elenco classificato delle sessioni di gioco che consumano la maggior parte delle risorse di memoria su questa istanza. I modelli di utilizzo elevato richiedono il controllo delle metriche delle singole sessioni e della distribuzione delle risorse. Fare clic sui collegamenti alle sessioni di gioco consente un'analisi più approfondita delle metriche dettagliate
	Le migliori sessioni di gioco che consumano la CPU	Elenco classificato delle sessioni di gioco che consumano la maggior parte delle risorse della CPU in questa istanza. I modelli di utilizzo elevato richiedono il controllo delle metriche delle singole sessioni e della distribuzione delle risorse come CPU e memoria. Facendo clic sui collegamenti alle sessioni di gioco è possibile eseguire un'analisi più approfondita delle metriche dettagliate
	Utilizzo della CPU per sessione di gioco	Analisi dell'utilizzo della CPU individuale che mostra il consumo di risorse per ogni sessione di gioco attiva
	Utilizzo della memoria per sessione di gioco	Analisi individuale dell'utilizzo della memoria che mostra il consumo di risorse per ogni sessione di gioco attiva
Medie attuali	Utilizzo della CPU dell'istanza	Utilizzo complessivo della CPU per l'istanza selezionata nel tempo
	Utilizzo della memoria dell'istanza	Utilizzo complessivo della memoria per l'istanza selezionata nel tempo

Pannello di controllo	Nome parametro	Definizione
	Utilizzo del file system dell'istanza	Consumo del file system per l'istanza selezionata, utile per identificare i problemi relativi alla capacità di storage
Rete	I/O di rete dell'istanza	La rete riflette la connettività delle istanze. Volume del traffico di rete e velocità dei pacchetti per l'istanza selezionata. Per problemi di latenza o connettività, esamina gli I/O di rete, la frequenza dei pacchetti e il conteggio degli errori
	Connessioni di rete	Numero di connessioni di rete attive sull'istanza selezionata
	Conteggi degli errori di rete	Statistiche sugli errori di rete per l'identificazione dei problemi di connettività
Disk	I/O Operazioni su disco	Il disco indica le prestazioni del disco. read/write Attività e utilizzo del disco per l'istanza selezionata. I tempi di risposta lenti suggeriscono di esaminare le operazioni su disco, i tempi di I/O attesa e le operazioni in sospeso
	Tempi di attesa del disco I/O	Tempo impiegato in attesa del completamento delle operazioni su disco
	Operazioni su disco in sospeso	Numero di operazioni su disco in attesa di essere elaborate

Note

Le tabelle Top N Memory and CPU Game Sessions sono essenziali per identificare i punti deboli in termini di prestazioni e i processi che richiedono molte risorse e che possono influire sulle prestazioni complessive dell'istanza. Queste classifiche consentono di identificare rapidamente le sessioni di gioco problematiche per ulteriori indagini.

Dashboard di panoramica sulla flotta di container

Questa dashboard offre una panoramica di alto livello dell'utilizzo delle risorse e della capacità della flotta di container. Visualizza l'utilizzo medio delle risorse di tutti i container in una flotta di container gestita, inclusa la prenotazione della CPU, l'utilizzo della memoria e lo stato del gruppo di container. Puoi filtrare per FleetID e Location.

Metriche di panoramica sulla flotta di container

La tabella seguente mostra le metriche disponibili nella dashboard di Container Fleet Overview:

Pannello di controllo	Nome parametro	Definizione
Stato del gruppo di container	Gruppi di contenitori Active Game Server	Gruppi di contenitori attualmente attivi e pronti per ospitare sessioni di gioco
	Gruppi di contenitori Idle Game Server	Gruppi di contenitori attivi ma che attualmente non ospitano sessioni di gioco
	Gruppi di contenitori di Game Server in sospeso	Gruppi di contenitori in fase di avvio
	Chiusura dei gruppi di contenitori di Game Server	Gruppi di contenitori in fase di chiusura
Utilizzo delle risorse	Utilizzo della CPU del contenitore	Utilizzo medio della CPU in tutti i container della flotta
	Utilizzo della memoria dei container	Utilizzo medio della memoria in tutti i container della flotta
	Prenotazione della CPU del contenitore	Percentuale di risorse CPU riservate dai container in tutta la flotta
Attività di rete	Container Network In	Byte di rete medi ricevuti dai container dell'intera flotta

Pannello di controllo	Nome parametro	Definizione
	Container Network Out	Byte di rete medi inviati dai container all'interno della flotta

Note

Le metriche della flotta di container vengono raccolte tramite ECS Container Receiver e forniscono una visibilità del carico di lavoro containerizzata con particolare attenzione all'aggregazione a livello di attività e all'isolamento delle risorse dei container.

Dashboard delle prestazioni dei container

Questa dashboard fornisce metriche dettagliate sulle prestazioni per i singoli contenitori all'interno di attività ECS specifiche. Visualizza metriche dettagliate dei singoli contenitori, tra cui l'utilizzo della CPU, l'utilizzo della memoria, l'I/O di rete e le prestazioni di archiviazione. È possibile filtrare per ID contenitore o attività ECS specifici.

Metriche relative alle prestazioni dei container

La tabella seguente mostra le metriche disponibili nella dashboard Container Performance:

Pannello di controllo	Nome parametro	Definizione
Prestazioni delle risorse	Utilizzo della CPU del contenitore	Utilizzo della CPU per il contenitore selezionato nel tempo
	Utilizzo della memoria del contenitore	Utilizzo della memoria per il contenitore selezionato nel tempo
	Prenotazione della memoria del contenitore	Percentuale di risorse di memoria riservate dal contenitore selezionato
Prestazioni di rete	I/O della rete di container	Volume di traffico di rete (byte in entrata/uscita) per il contenitore selezionato

Pannello di controllo	Nome parametro	Definizione
	Connessioni di rete	Numero di connessioni di rete attive per il contenitore selezionato
Prestazioni di archiviazione	Lettura/scrittura dello storage in container	Attività di archiviazione, lettura e scrittura per il contenitore selezionato
	I/O Operazioni di stoccaggio dei container	Numero di I/O operazioni di archiviazione eseguite dal contenitore selezionato

Note

Le metriche delle prestazioni dei container forniscono una visibilità dettagliata sul consumo di risorse dei singoli container e sulle caratteristiche prestazionali all'interno delle attività ECS.

Dashboard delle prestazioni del server

La dashboard Server Performance mostra le metriche relative ai tempi del server, all'attività di rete, alla memoria e all'utilizzo della CPU per le singole sessioni di gioco. Puoi filtrare in base all'ID della sessione di gioco ed esportare le metriche direttamente su Amazon CloudWatch o [Amazon Managed Grafana](#).

Metriche sulle prestazioni del server

La tabella seguente mostra le metriche disponibili nel pannello di controllo delle prestazioni del server:

Pannello di controllo	Nome parametro	Definizione
Orari del server	Delta Time del server	Differenza di tempo tra il segno di spunta corrente del server e il segno di spunta del server precedente. È una misura della coerenza del tick rate del server

Pannello di controllo	Nome parametro	Definizione
	Tick Rate del server	Mostra il numero di volte al secondo in cui il server elabora gli aggiornamenti
	Orario di tick-time del server	Il tempo impiegato dal server per elaborare un singolo segno di spunta o aggiornamento
	Server World Tick Time	Il tempo impiegato dal server per aggiornare il mondo di gioco con ogni segno di spunta
Rete	Connessioni	Il numero totale di connessioni di rete stabilite dal server
	Rete I/O (byte)	Il numero totale di byte inviati e ricevuti dal server sulla rete
	Rete I/O (pacchetti)	Il numero totale di pacchetti di rete inviati e ricevuti dal server
	Perdita di pacchetti	La percentuale di pacchetti di rete che vengono persi durante la trasmissione
Memoria	Utilizzo della memoria (unità)	La quantità totale di memoria consumata dai processi del server
	Utilizzo della memoria fisica (%)	La percentuale della memoria fisica totale del server attualmente utilizzata
Utilizzo CPU	Utilizzo della CPU (%)	La percentuale di risorse della CPU utilizzate dal processo del server di gioco

Scenari di monitoraggio comuni

Approfondisci le indagini sulle prestazioni

Scenario: A host/instance presenta prestazioni ridotte a causa di processi o sessioni di gioco specifici

Fasi dell'indagine:

- Accedi alla dashboard delle prestazioni delle istanze.
- Consulta la tabella «Le migliori N sessioni di gioco che consumano memoria» per identificare quali processi contribuiscono maggiormente al consumo di memoria delle istanze.
- Consulta la tabella «Le N principali sessioni di gioco che consumano CPU» per identificare quali processi contribuiscono maggiormente all'utilizzo della CPU dell'istanza.
- Fai clic sui link delle sessioni di gioco per consentire un'analisi più approfondita delle metriche dettagliate.
- Analizza gli orari dei server (Server Delta Time, Server Tick Rate, Server Tick Time, Server World Tick Time) per identificare i punti deboli nelle prestazioni.

Indagine sugli incidenti dei server di gioco

Scenario: una sessione di gioco si è bloccata ed è necessario determinarne la causa principale

Fasi dell'indagine:

- Accedi alla dashboard delle prestazioni del server per la sessione di gioco interrotta.
- Controlla l'utilizzo della memoria (unità) e l'utilizzo della memoria fisica (%) per determinare se l'arresto anomalo è dovuto all'esaurimento della memoria.
- Controlla l'utilizzo della CPU (%) per identificare se il crash è stato causato dal sovraccarico della CPU.
- Analizza la rete I/O (byte) e la rete I/O (pacchetti) per determinare se i problemi di larghezza di banda della rete hanno contribuito al crash.
- Esamina la percentuale di perdita di pacchetti per identificare i problemi relativi alla rete.

Esamina i problemi segnalati dai giocatori

Scenario: i giocatori segnalano ritardi o interruzioni durante il gioco

Fasi dell'indagine:

- Accedi alla dashboard delle prestazioni del server per la sessione di gioco interessata.
- Consulta Server Tick Time e Server World Tick Time per identificare i ritardi negli aggiornamenti del gioco.
- Controlla Server Tick Rate per garantire una frequenza di aggiornamento costante del server.

- Analizza l'utilizzo della CPU (%) per identificare i colli di bottiglia nell'elaborazione.
- Rivedi le metriche di utilizzo della memoria per identificare i problemi di prestazioni relativi alla memoria.
- Controlla le I/O metriche di rete e la perdita di pacchetti per identificare i punti deboli della rete.

Identifica le variazioni delle prestazioni nelle diverse build di server di gioco

Scenario: vuoi misurare come cambiano le prestazioni di gioco tra diverse build di server

Fasi dell'indagine:

- Confronta le metriche di Server Tick Time tra diverse build per misurare le variazioni dell'efficienza di elaborazione.
- Analizza la coerenza del Server Tick Rate tra le build per identificare le regressioni delle prestazioni.
- Consulta Server World Tick Time per misurare i cambiamenti nelle prestazioni degli aggiornamenti del mondo di gioco.
- Confronta i modelli di utilizzo della memoria tra le build per identificare miglioramenti o regressioni dell'ottimizzazione della memoria.
- Monitora le tendenze di utilizzo della CPU per valutare le variazioni dell'efficienza computazionale.

Rileva ritardi e rallentamenti nel gameplay

Scenario: devi monitorare la reattività del server e la velocità di aggiornamento del gioco

Fasi dell'indagine:

- Monitora il Tick Time del server per misurare la velocità con cui il server elabora ogni ciclo di aggiornamento.
- Tieni traccia della frequenza di selezione del server per garantire aggiornamenti costanti allo stato del gioco al secondo.
- Analizza Server World Tick Time per misurare la velocità di aggiornamento del mondo di gioco, che influisce direttamente sull'esperienza del cliente.
- Imposta avvisi per le variazioni del Delta Time del server per rilevare prestazioni incoerenti del server.

Analisi comparativa di diversi scenari di gioco

Scenario: vuoi identificare in che modo i diversi scenari di gioco influiscono sulle prestazioni del server

Fasi dell'indagine:

- Confronta le metriche delle prestazioni del server in base al numero di giocatori diversi per comprendere l'impatto della scalabilità.
- Analizza le differenze di prestazioni tra le modalità di gioco utilizzando le metriche Server Tick Time e CPU Usage.
- Monitora i modelli di utilizzo della memoria in diversi scenari di gioco per identificare le funzionalità che richiedono molte risorse.
- Tieni traccia delle I/O metriche di rete per comprendere i requisiti di larghezza di banda per diversi scenari di gioco.
- Usa la dashboard Instance Performance per identificare quali scenari di gioco producono le sessioni di gioco che consumano più risorse.

Risposta a un elevato utilizzo delle risorse

Scenario: picchi di risorse insoliti (CPU > 85%, memoria > 90%)

Fasi dell'indagine:

Identifica le risorse interessate

- Usa l' `DescribeGameSessionDetails` API.
- Filtra per stato, se necessario.
- Documenta le istanze interessate.

Analizza l'utilizzo delle risorse

- Consulta la dashboard di panoramica delle istanze.
- Confronta l'utilizzo in tutta la flotta.
- Controlla i modelli storici.

Monitora l'impatto sui server di gioco

- Controlla le metriche delle prestazioni del server.
- Controlla i tempi di verifica e la perdita di pacchetti.
- Monitora le perdite di memoria.

Passaggi di risoluzione

- Scarica i registri delle sessioni.
- Risolvi i problemi di compilazione.
- Monitora i miglioramenti.

Analisi dei crash del server di gioco

Scenario: più sessioni di gioco con stato di errore in tutta la flotta

Fasi dell'indagine:

Valutazione iniziale

- Accedi alla dashboard di panoramica della flotta.
- Rivedi la tabella delle sessioni bloccate.
- Annota gli schemi in termini di tempistica/luogo.

Analisi delle prestazioni

- Controlla le metriche di temporizzazione del server.
- Esamina l'utilizzo delle risorse.
- Monitora le prestazioni della rete.

Revisione dell'infrastruttura

- Verifica la capacità della flotta.
- Controlla lo stato dell'istanza.
- Rivedi le politiche di scalabilità.

Percorso di risoluzione

- Analizza i log del server.
- Ottimizzazione del codice di revisione.
- Implementa correzioni.

Ottimizzazione della capacità della flotta

Scenario: lancio del gioco o studio di benchmark

Fasi dell'analisi:

Utilizzo delle risorse

- Filtra per località.
- Esamina le metriche P50/P95/P99.
- Analizza i modelli di utilizzo.

Analisi del tipo di istanza

- Confronta le prestazioni per tipo.
- Identifica candidati scalabili.
- Modelli di utilizzo dei documenti.

Azioni di ottimizzazione

- Modifica le politiche di scalabilità.
- Modifica i tipi di istanza.
- Aggiorna la configurazione del parco veicoli.

Guida alla risoluzione dei problemi

Problemi comuni e passaggi di risoluzione

Metriche mancanti o incomplete

Caratteristiche

- Nessuna metrica visualizzata nei dashboard.
- Raccolta metrica parziale.
- Aggiornamenti delle metriche ritardati.

Passaggi di risoluzione

A. Verifica lo stato del raccoglitore

Controlla il servizio systemd:

```
sudo systemctl status gamelift-telemetry-collector
```

Rivedi i log di Collector:

```
sudo journalctl -u gamelift-telemetry-collector
```

- Conferma la configurazione del raccoglitore.

B. Verifica dell'autorizzazione IAM

- Controlla le autorizzazioni del ruolo dell'istanza.
- Verifica le politiche richieste:
 - `aps:RemoteWrite`
 - `cloudwatch:PutMetricData`
- Convalida le relazioni di fiducia tra i ruoli.

C. Connettività di rete

- Verificare l'accesso agli endpoint.

- Controlla le regole del gruppo di sicurezza.
- Rivedi la rete ACLs.

Errori di autenticazione

Caratteristiche

- Errori di autenticazione SigV4.
- Accesso negato ai messaggi.
- Problemi di aggiornamento delle credenziali.

Passaggi di risoluzione

A. Autenticazione SigV4

- Verifica le credenziali temporanee.
- Controlla la rotazione delle credenziali.
- Convalida il profilo dell'istanza.

B. Accesso AMP

- Rivedi la configurazione dell'area di lavoro.
- Verifica l'URL di scrittura remota.
- Controlla le associazioni di ruolo IAM.

Problemi con la dashboard

Caratteristiche

- Dashboard vuoti.
- Punti dati mancanti.
- Errori di autenticazione.

Passaggi di risoluzione

A. Configurazione della fonte di dati

- Verifica la connessione con Prometheus.
- Verifica CloudWatch l'integrazione con Amazon.
- Verifica le autorizzazioni delle fonti di dati.

B. Accesso a Grafana

- Conferma la configurazione SSO.
- Se necessario, verifica la configurazione 2FA.
- Controlla le autorizzazioni degli utenti.

Problemi specifici di Windows

Caratteristiche

- Errori di avvio del servizio.
- Lacune di raccolta metriche.
- Errori di autorizzazione.

Passaggi di risoluzione

- Verifica lo stato del servizio Windows.
- Controlla i registri degli eventi di Windows.
- Controlla la configurazione del raccoglitore.
- Convalida i percorsi specifici di Windows.

Registrazione delle chiamate Amazon GameLift Servers API con AWS CloudTrail

Amazon GameLift Servers è integrato con AWS CloudTrail, un servizio che fornisce una registrazione delle azioni intraprese da un utente, un ruolo o un AWS servizio in Amazon GameLift Servers.

CloudTrail acquisisce tutte le chiamate API Amazon GameLift Servers come eventi. Le chiamate acquisite includono le chiamate dalla console di Amazon GameLift Servers e le chiamate di codice alle operazioni delle API Amazon GameLift Servers. Se crei un trail, puoi abilitare la distribuzione continua di CloudTrail eventi a un bucket Amazon S3, inclusi gli eventi per Amazon GameLift Servers. Se non configuri un percorso, puoi comunque visualizzare gli eventi più recenti nella CloudTrail console nella cronologia degli eventi. Utilizzando le informazioni raccolte da CloudTrail, puoi determinare a quale richiesta è stata inviata Amazon GameLift Servers, l'indirizzo IP da cui è stata effettuata la richiesta, chi ha effettuato la richiesta, quando è stata effettuata e dettagli aggiuntivi.

Per ulteriori informazioni CloudTrail, consulta la [Guida AWS CloudTrail per l'utente](#).

Amazon GameLift Servers informazioni in CloudTrail

CloudTrail è abilitato sul tuo account al momento della creazione dell'account. Quando si verifica un'attività in Amazon GameLift Servers, tale attività viene registrata in un CloudTrail evento insieme ad altri eventi AWS di servizio nella cronologia degli eventi. Puoi visualizzare, cercare e scaricare eventi recenti in Account AWS. Per ulteriori informazioni, consulta [Visualizzazione degli eventi con la cronologia degli CloudTrail eventi](#).

Per una registrazione continua degli eventi nell' Account AWS che includa gli eventi per Amazon GameLift Servers, crea un trail. Un trail consente di CloudTrail inviare file di log a un bucket Amazon S3. Per impostazione predefinita, quando si crea un percorso nella console, questo sarà valido in tutte le Regioni AWS. Il trail registra gli eventi di tutte le regioni della AWS partizione e consegna i file di log al bucket Amazon S3 specificato. Inoltre, puoi configurare altri AWS servizi per analizzare ulteriormente e agire in base ai dati sugli eventi raccolti nei log. CloudTrail Per ulteriori informazioni, consulta gli argomenti seguenti:

- [Panoramica della creazione di un percorso](#)
- [CloudTrail servizi e integrazioni supportati](#)
- [Configurazione delle notifiche Amazon SNS per CloudTrail](#)
- [Ricezione di file di CloudTrail registro da più regioni](#) e [ricezione di file di CloudTrail registro da più account](#)

Tutte Amazon GameLift Servers le azioni vengono registrate CloudTrail e documentate nell'[Amazon GameLift Servers API Reference](#). Ad esempio, le chiamate `CreatePlayerSession` e `UpdateGameSession` le azioni generano voci nei file di CloudTrail registro. `CreateGameSession`

Ogni evento o voce di log contiene informazioni sull'utente che ha generato la richiesta. Le informazioni di identità consentono di determinare quanto segue:

- Se la richiesta è stata effettuata con credenziali utente root o AWS Identity and Access Management (IAM).
- Se la richiesta è stata effettuata con le credenziali di sicurezza temporanee per un ruolo o un utente federato.
- Se la richiesta è stata effettuata da un altro AWS servizio.

Per ulteriori informazioni, consulta [Elemento CloudTrail userIdentity](#).

Informazioni sulle voci dei file di log di Amazon GameLift Servers

Un trail è una configurazione che consente la distribuzione di eventi come file di log in un bucket Amazon S3 specificato dall'utente. CloudTrail i file di registro contengono una o più voci di registro. Un evento rappresenta una singola richiesta proveniente da qualsiasi fonte e include informazioni sull'azione richiesta, la data e l'ora dell'azione, i parametri della richiesta e così via. CloudTrail i file di registro non sono una traccia ordinata dello stack delle chiamate API pubbliche, quindi non vengono visualizzati in un ordine specifico.

L'esempio seguente mostra una voce di CloudTrail registro che illustra le azioni `CreateFleet` e `DescribeFleetAttributes`.

```
{
  "Records": [
    {
      "eventVersion": "1.04",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/myUserName",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "myUserName"
      },
      "eventTime": "2015-12-29T23:40:15Z",
      "eventSource": "gamelift.amazonaws.com",
      "eventName": "CreateFleet",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "192.0.2.0",
```

```

    "userAgent": "[]",
    "requestParameters": {
      "buildId": "build-92b6e8af-37a2-4c10-93bd-4698ea23de8d",
      "eC2InboundPermissions": [
        {
          "ipRange": "10.24.34.0/23",
          "fromPort": 1935,
          "protocol": "TCP",
          "toPort": 1935
        }
      ],
      "logPaths": [
        "C:\\\\game\\\\serverErr.log",
        "C:\\\\game\\\\serverOut.log"
      ],
      "eC2InstanceType": "c5.large",
      "serverLaunchPath": "C:\\\\game\\\\MyServer.exe",
      "description": "Test fleet",
      "serverLaunchParameters": "-paramX=baz",
      "name": "My_Test_Server_Fleet"
    },
    "responseElements": {
      "fleetAttributes": {
        "fleetId": "fleet-0bb84136-4f69-4bb2-bfec-a9b9a7c3d52e",
        "serverLaunchPath": "C:\\\\game\\\\MyServer.exe",
        "status": "NEW",
        "logPaths": [
          "C:\\\\game\\\\serverErr.log",
          "C:\\\\game\\\\serverOut.log"
        ],
        "description": "Test fleet",
        "serverLaunchParameters": "-paramX=baz",
        "creationTime": "Dec 29, 2015 11:40:14 PM",
        "name": "My_Test_Server_Fleet",
        "buildId": "build-92b6e8af-37a2-4c10-93bd-4698ea23de8d"
      }
    },
    "requestID": "824a2a4b-ae85-11e5-a8d6-61d5cafb25f2",
    "eventID": "c8fbea01-fbf9-4c4e-a0fe-ad7dc205ce11",
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
  },
  {
    "eventVersion": "1.04",

```

```

    "userIdentity": {
      "type": "IAMUser",
      "principalId": "AIDACKCEVSQ6C2EXAMPLE",
      "arn": "arn:aws:iam::111122223333:user/myUserName",
      "accountId": "111122223333",
      "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
      "userName": "myUserName"
    },
    "eventTime": "2015-12-29T23:40:15Z",
    "eventSource": "gamelift.amazonaws.com",
    "eventName": "DescribeFleetAttributes",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "[]",
    "requestParameters": {
      "fleetIds": [
        "fleet-0bb84136-4f69-4bb2-bfec-a9b9a7c3d52e"
      ]
    },
    "responseElements": null,
    "requestID": "82e7f0ec-ae85-11e5-a8d6-61d5cafb25f2",
    "eventID": "11daabcb-0094-49f2-8b3d-3a63c8bad86f",
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
  },
]
}

```

Registrazione dei messaggi del server Amazon GameLift Servers

Puoi acquisire messaggi server personalizzati dai tuoi Amazon GameLift Servers server in file di registro. Il modo in cui configuri la registrazione dipende dal fatto che utilizzi server personalizzati o Amazon GameLift Servers Realtime che utilizzi EC2 flotte gestite o flotte di container (vedi le sottosezioni appropriate in questo capitolo).

Argomenti

- [Differenze di registrazione: flotte gestite EC2 e flotte container](#)
- [Messaggi del server di registrazione \(server personalizzati\)](#)

Differenze di registrazione: flotte gestite EC2 e flotte container

Il comportamento di registrazione differisce in modo significativo tra flotte gestite EC2 e flotte di container. Comprendere queste differenze è fondamentale quando si passa dai container EC2 ai container o si sceglie il tipo di flotta giusto per le proprie esigenze di registrazione.

Panoramica delle differenze principali

Funzionalità	EC2 Flotte gestite	Flotte di container
Archiviazione dei registri	Caricato automaticamente su Amazon S3 al termine della sessione di gioco	È necessario configurare Amazon CloudWatch Logs o una soluzione di registrazione personalizzata
GetGameSessionLogUrl API	Disponibile: restituisce l'URL di Amazon S3 per il download del log	Non disponibile: i log non vengono archiviati automaticamente in Amazon S3
Retention dei log	14 giorni in Amazon S3 (automatico)	Dipende dalla configurazione di registrazione
monitoraggio in tempo reale	Limitato: i registri sono disponibili solo dopo la fine della sessione di gioco	Disponibile con l'integrazione con Amazon CloudWatch Logs
Complessità della configurazione	Automatico: non è richiesta alcuna configurazione aggiuntiva	Richiede una configurazione di registrazione esplicita

Registrazione gestita EC2 del parco veicoli

Per le EC2 flotte gestite, Amazon GameLift Servers fornisce la gestione automatica dei registri:

- Caricamento automatico: i log del server vengono caricati automaticamente su Amazon S3 al termine di una sessione di gioco
- GetGameSessionLogUrl API: utilizza questa API per recuperare un URL prefirmato per scaricare i log da Amazon S3
- Conservazione per 14 giorni: i log vengono conservati in Amazon S3 per 14 giorni prima dell'eliminazione automatica

- Limiti di dimensione: [i file di registro hanno limiti di dimensione per sessione di gioco \(vedi endpoint e quote\) Amazon GameLift Servers](#)

Per ulteriori informazioni, consulta [the section called “Registrazione per server personalizzati”](#) e il riferimento all'[GetGameSessionLogUrl API](#).

Registrazione della flotta di container

Le flotte di container richiedono di configurare la registrazione in modo esplicito:

- Nessun caricamento automatico di Amazon S3: i log non vengono caricati automaticamente su Amazon S3
- GetGameSessionLogUrl non disponibile: questa API non funziona con flotte di container
- Integrazione con Amazon CloudWatch Logs: configura il contenitore per inviare log ad Amazon CloudWatch Logs per la registrazione centralizzata
- Soluzioni di registrazione personalizzate: implementa la tua infrastruttura di registrazione utilizzando driver di log o contenitori collaterali
- Accesso in tempo reale: con una configurazione corretta, è possibile accedere ai log in tempo reale durante le sessioni di gioco

Per le opzioni dettagliate di registrazione dei container, consulta [Come funzionano le flotte di container](#).

Considerazioni sulla migrazione

Quando esegui la migrazione da flotte gestite EC2 a flotte di container, considera queste modifiche alla registrazione:

- Aggiorna il codice di recupero dei log: sostituisci le chiamate GetGameSessionLogUrl API con le query Amazon CloudWatch Logs o la tua soluzione di registrazione personalizzata
- Configurazione della conservazione dei log: configura politiche di conservazione appropriate in Amazon CloudWatch Logs o nel tuo sistema di registrazione
- Implementa il monitoraggio in tempo reale: sfrutta l'accesso ai log in tempo reale per una migliore osservabilità
- Esamina il volume e i costi dei log: CloudWatch i prezzi di Amazon Logs differiscono dallo storage Amazon S3 incluso nelle flotte gestite EC2

Configurazione consigliata per la registrazione dei container

Per le flotte di container, consigliamo:

1. Integrazione con Amazon CloudWatch Logs: configura la definizione del contenitore per utilizzare il driver di `awslogs` registro
2. Registrazione strutturata: utilizza formati di log strutturati (JSON) per una migliore ricercabilità e analisi
3. Livelli di registro: implementa livelli di registro appropriati per controllare la verbosità e i costi
4. Politiche di conservazione: imposta i periodi di conservazione in base alle tue esigenze di conformità e debug

Messaggi del server di registrazione (server personalizzati)

È possibile acquisire messaggi server personalizzati dai server Amazon GameLift Servers personalizzati in file di registro.

Important

C'è un limite alla dimensione di un file di registro per sessione di gioco (vedi [Amazon GameLift Servers endpoint e quote](#) in). Riferimenti generali di AWS Al termine di una sessione di gioco, Amazon GameLift Servers carica i log del server su Amazon Simple Storage Service (Amazon S3). Amazon GameLift Servers non caricherà log che superano il limite. I log possono crescere molto rapidamente e superare il limite di dimensione. È necessario monitorare i log e limitare l'output dei log ai soli messaggi necessari.

Configurazione della registrazione per server personalizzati

Con i server Amazon GameLift Servers personalizzati, è possibile scrivere il proprio codice per eseguire la registrazione, che viene configurato come parte della configurazione del processo del server. Amazon GameLift Servers utilizza la tua configurazione di registrazione per identificare i file che deve caricare su S3 alla fine di ogni sessione di gioco.

Le seguenti istruzioni mostrano come configurare la registrazione utilizzando esempi di codice semplificati:

C++

Per configurare la registrazione (C++)

1. Crea un vettore di stringhe che siano percorsi di directory verso i file di registro del server di gioco.

```
std::string serverLog("serverOut.log");           // Example server log file
std::vector<std::string> logPaths;
logPaths.push_back(serverLog);
```

2. Fornisci il tuo vettore come immagine [LogParameters](#) del tuo oggetto. [ProcessParameters](#)

```
Aws::GameLift::Server::ProcessParameters processReadyParameter =
    Aws::GameLift::Server::ProcessParameters(
        std::bind(&Server::onStartGameSession, this, std::placeholders::_1),
        std::bind(&Server::onProcessTerminate, this),
        std::bind(&Server::OnHealthCheck, this),
        std::bind(&Server::OnUpdateGameSession, this),
        listenPort,
        Aws::GameLift::Server::LogParameters(logPaths);
```

3. Fornisci l'[ProcessParameters](#) oggetto quando chiami [ProcessReady\(\)](#).

```
Aws::GameLift::GenericOutcome outcome =
    Aws::GameLift::Server::ProcessReady(processReadyParameter);
```

Per un esempio più completo, vedi [ProcessReady\(\)](#).

C#

Per configurare la registrazione (C#)

1. Crea un elenco di stringhe che sono percorsi di directory verso i file di registro del server di gioco.

```
List<string> logPaths = new List<string>();
logPaths.Add("C:\\game\\serverOut.txt");           // Example of a log file that the
game server writes
```

2. Fornisci la tua lista come [ProcessParameters](#) oggetto. [LogParameters](#)

```
var processReadyParameter = new ProcessParameters(  
    this.OnGameSession,  
    this.OnProcessTerminate,  
    this.OnHealthCheck,  
    this.OnGameSessionUpdate,  
    port,  
    new LogParameters(logPaths));
```

3. Fornisci l'[ProcessParameters](#) oggetto quando chiami [ProcessReady\(\)](#).

```
var processReadyOutcome =  
    GameLiftServerAPI.ProcessReady(processReadyParameter);
```

Per un esempio più completo, vedi [ProcessReady\(\)](#).

Scrittura nei log

I file di registro esistono dopo l'avvio del processo del server. È possibile scrivere nei log utilizzando qualsiasi metodo di scrittura sui file. Per acquisire tutti gli output standard e gli output di errore del server, rimappa i flussi di output nei file di registro, come negli esempi seguenti:

C++

```
std::freopen("serverOut.log", "w+", stdout);  
std::freopen("serverErr.log", "w+", stderr);
```

C#

```
Console.SetOut(new StreamWriter("serverOut.txt"));  
Console.SetError(new StreamWriter("serverErr.txt"));
```

Accesso ai log del server

L'accesso ai log varia in base al tipo di parco veicoli:

Flotte EC2 gestite

Al termine di una sessione di gioco, archivia Amazon GameLift Servers automaticamente i log in un bucket S3 e li conserva per 14 giorni. Per ottenere la posizione dei log di una sessione di gioco, puoi utilizzare l'operazione API. [GetGameSessionLogUrl](#) Per scaricare i log, utilizza l'URL restituito dall'operazione.

In alternativa, puoi configurare la tua soluzione di registrazione configurando il tuo server di gioco in modo che invii i log direttamente al servizio di registrazione o alla posizione di archiviazione preferita. Per ulteriori informazioni, consulta [Connect il server di gioco Amazon GameLift Servers ospitato ad altre AWS risorse](#).

Flotte di container

Le flotte di container acquisiscono flussi di output e di errore standard da tutti i container. È possibile configurare la registrazione utilizzando una di queste opzioni:

- Salva l'output del contenitore come flussi di CloudWatch log in un gruppo di log specificato
- Salva l'output del contenitore in un bucket di archiviazione S3
- Disattiva la registrazione (l'output del contenitore non viene salvato)

Per informazioni dettagliate sulla configurazione delle opzioni di registrazione durante la creazione di flotte di container, consulta [LogConfiguration](#) l'API Reference. Amazon GameLift Servers

Sicurezza in Amazon GameLift Servers

Se la utilizzi Amazon GameLift Servers FleetIQ come funzionalità autonoma con Amazon EC2, [consulta la sezione Sicurezza in Amazon EC2 nella Guida per l'utente di Amazon EC2](#).

La sicurezza del cloud AWS è la massima priorità. In quanto cliente AWS, è possibile trarre vantaggio da un'architettura di data center e di rete progettata per soddisfare i requisiti delle aziende più esigenti a livello di sicurezza.

La sicurezza è una responsabilità condivisa tra te AWS e te. Il [modello di responsabilità condivisa](#) descrive questo aspetto come sicurezza del cloud e sicurezza nel cloud:

- Sicurezza del cloud: AWS è responsabile della protezione dell'infrastruttura che gestisce AWS i servizi nel AWS cloud. AWS ti fornisce anche servizi che puoi utilizzare in modo sicuro. Per maggiori informazioni sui programmi di conformità applicabili a Amazon GameLift Servers, consulta la sezione [AWS Servizi rientranti nell'ambito dei programmi di conformità](#) ().
- Sicurezza nel cloud: la tua responsabilità è determinata dal AWS servizio che utilizzi. Sei responsabile anche di altri fattori, tra cui la sensibilità dei tuoi dati, i requisiti aziendali AWS e le normative applicabili.

Questa documentazione facilita la comprensione su come applicare il modello di responsabilità condivisa quando si utilizza Amazon GameLift Servers. I seguenti argomenti illustrano come configurare l'Amazon GameLift Servers per soddisfare gli obiettivi di sicurezza e conformità. Imparerai anche a utilizzare altri AWS servizi che ti aiutano a monitorare e proteggere Amazon GameLift Servers le tue risorse.

Argomenti

- [Protezione dei dati in Amazon GameLift Servers](#)
- [Gestione delle identità e degli accessi per l' Amazon GameLift Servers](#)
- [Registrazione e monitoraggio con Amazon GameLift Servers](#)
- [Convalida della conformità per Amazon GameLift Servers](#)
- [Resilienza in Amazon GameLift Servers](#)
- [Sicurezza dell'infrastruttura in Amazon GameLift Servers](#)
- [Analisi della configurazione e delle vulnerabilità in Amazon GameLift Servers](#)
- [Best practice relative alla sicurezza di Amazon GameLift Servers](#)

Protezione dei dati in Amazon GameLift Servers

Se utilizzi Amazon GameLift Servers FleetIQ come funzionalità autonoma con Amazon EC2, consulta la sezione [Security in Amazon EC2 nella Amazon EC2 User Guide](#).

Il modello di [responsabilità AWS condivisa modello](#) si applica alla protezione dei dati in Amazon GameLift Servers. Come descritto in questo modello, AWS è responsabile della protezione dell'infrastruttura globale che gestisce tutti i Cloud AWS. L'utente è responsabile del controllo dei contenuti ospitati su questa infrastruttura. L'utente è inoltre responsabile della configurazione della protezione e delle attività di gestione per i Servizi AWS utilizzati. Per maggiori informazioni sulla privacy dei dati, consulta le [Domande frequenti sulla privacy dei dati](#). Per informazioni sulla protezione dei dati in Europa, consulta il post del blog relativo al [AWS Modello di responsabilità condivisa e GDPR](#) nel AWS Blog sulla sicurezza.

Ai fini della protezione dei dati, consigliamo di proteggere Account AWS le credenziali e configurare i singoli utenti con AWS IAM Identity Center o AWS Identity and Access Management (IAM). In tal modo, a ogni utente verranno assegnate solo le autorizzazioni necessarie per svolgere i suoi compiti. Suggeriamo, inoltre, di proteggere i dati nei seguenti modi:

- Utilizza l'autenticazione a più fattori (MFA) con ogni account.
- SSL/TLS Da utilizzare per comunicare con AWS le risorse. È richiesto TLS 1.2 ed è consigliato TLS 1.3.
- Configura l'API e la registrazione delle attività degli utenti con AWS CloudTrail. Per informazioni sull'utilizzo dei CloudTrail percorsi per acquisire AWS le attività, consulta [Lavorare con i CloudTrail percorsi](#) nella Guida per l'AWS CloudTrail utente.
- Utilizza soluzioni di AWS crittografia, insieme a tutti i controlli di sicurezza predefiniti all'interno Servizi AWS.
- Utilizza i servizi di sicurezza gestiti avanzati, come Amazon Macie, che aiutano a individuare e proteggere i dati sensibili archiviati in Amazon S3.
- Se hai bisogno di moduli crittografici convalidati FIPS 140-3 per accedere AWS tramite un'interfaccia a riga di comando o un'API, usa un endpoint FIPS. Per ulteriori informazioni sugli endpoint FIPS disponibili, consulta il [Federal Information Processing Standard \(FIPS\) 140-3](#).

Ti consigliamo di non inserire mai informazioni riservate o sensibili, ad esempio gli indirizzi e-mail dei clienti, nei tag o nei campi di testo in formato libero, ad esempio nel campo Nome. Ciò include quando lavori Amazon GameLift Servers o Servizi AWS utilizzi la console, l'API o. AWS CLI AWS

SDKs I dati inseriti nei tag o nei campi di testo in formato libero utilizzati per i nomi possono essere utilizzati per la fatturazione o i log di diagnostica. Quando si fornisce un URL a un server esterno, suggeriamo vivamente di non includere informazioni sulle credenziali nell'URL per convalidare la richiesta al server.

I dati specifici di Amazon GameLift Servers vengono gestiti come segue:

- Le build dei server di gioco e gli script su cui carichi Amazon GameLift Servers vengono archiviati in Amazon S3. Non vi è alcun accesso diretto del cliente a questi dati una volta caricati. Un utente autorizzato può ottenere l'accesso temporaneo al caricamento dei file, ma non può visualizzare o aggiornare i file direttamente in Amazon S3. Per eliminare script e build, utilizza la console Amazon GameLift Servers o l'API del servizio.
- I dati di registro delle sessioni di gioco vengono archiviati in Amazon S3 per un periodo di tempo limitato dopo il completamento della sessione di gioco. Gli utenti autorizzati possono accedere ai dati di log scaricandoli tramite un collegamento nella console Amazon GameLift Servers o tramite chiamate all'API del servizio.
- I dati relativi a parametri ed eventi vengono archiviati in Amazon GameLift Servers ed è possibile accedervi tramite la console Amazon GameLift Servers o le chiamate all'API del servizio. I dati possono essere recuperati su parchi istanze, istanze, posizionamenti di sessioni di gioco, ticket di matchmaking, sessioni di gioco e sessioni di giocatori. È possibile accedere ai dati anche tramite Amazon CloudWatch ed CloudWatch Events.
- I dati forniti dal cliente vengono memorizzati in Amazon GameLift Servers. Gli utenti autorizzati possono accedervi tramite chiamate all'API del servizio. I dati potenzialmente sensibili possono includere i dati del giocatore, i dati della sessione del giocatore e della sessione di gioco (incluse le informazioni sulla connessione), i dati di matchmaker e così via.

Note

Se inserisci un giocatore personalizzato IDs nelle tue richieste, è previsto che questi valori siano resi anonimi UUIDs e non contengano informazioni identificative del giocatore.

Per ulteriori informazioni sulla protezione dei dati, consulta il post del blog relativo al [modello di responsabilità condivisa AWS e GDPR](#) in AWS Security Blog.

Crittografia dei dati a riposo

La crittografia dei dati inattivi specifici di Amazon GameLift Servers viene gestita come segue:

- Le build e gli script dei server di gioco sono archiviati in bucket Amazon S3 con crittografia lato server.
- I dati forniti dal cliente vengono archiviati in Amazon GameLift Servers in formato crittografato.

Crittografia dei dati in transito

Le connessioni a Amazon GameLift Servers APIs vengono effettuate tramite una connessione sicura (SSL) e autenticate utilizzando la [versione 4 di AWS Signature](#) (quando ci si connette tramite AWS CLI o AWS SDK, la firma viene gestita automaticamente). L'autenticazione viene gestita utilizzando le policy di accesso definite da IAM per le credenziali di sicurezza utilizzate per effettuare la connessione.

La comunicazione diretta tra i client di gioco e i server di gioco è la seguente:

- Per i server di gioco personalizzati ospitati in risorse Amazon GameLift Servers, la comunicazione non coinvolge il servizio Amazon GameLift Servers. La crittografia di questa comunicazione è una responsabilità del cliente. È possibile utilizzare parchi istanze abilitati per TLS per consentire ai client di gioco di autenticare il server di gioco durante la connessione e per crittografare tutte le comunicazioni tra il client di gioco e il server di gioco.
- I certificati per le flotte compatibili con TLS vengono creati contemporaneamente al parco veicoli e le relative date di scadenza si basano sulla data di creazione del parco veicoli.
- Infatti, Amazon GameLift Servers Realtime con la generazione di certificati TLS abilitata, il traffico tra il client di gioco e Realtime i server che utilizza il client SDK for Realtime viene crittografato durante il trasferimento. Il traffico TCP viene crittografato con TLS 1.2 e il traffico UDP viene crittografato con DTLS 1.2.

Note

Per soddisfare i requisiti dell'Autorità di certificazione per i certificati TLS, Amazon GameLift Servers modificherà la durata massima dei certificati per le flotte configurate per generare un certificato. La durata dei certificati inizia al momento della creazione della flotta e verrà modificata secondo la seguente pianificazione:

- Fino all'11 marzo 2026, la durata massima di un certificato TLS emesso è di 398 giorni.
- A partire dal 1° marzo 2026, la durata massima di un certificato TLS emesso sarà di 200 giorni.

- A partire dal 1° marzo 2027, la durata massima di un certificato TLS emesso sarà di 100 giorni.
- A partire dal 1° marzo 2029, la durata massima di un certificato TLS emesso sarà di 47 giorni.

Per garantire il rinnovo dei certificati e per mantenere gli ambienti di runtime dei server di up-to-date gioco, consigliamo di sostituire regolarmente Amazon GameLift Servers le flotte dei server di gioco.

Riservatezza del traffico Internet

È possibile accedere in remoto alle istanze Amazon GameLift Servers in modo sicuro. Per le istanze che utilizzano Linux, SSH fornisce un canale di comunicazione sicuro per l'accesso remoto. Per le istanze che eseguono Windows, utilizza un client RDP (Remote Desktop Protocol). Con Amazon GameLift ServersFleetIQ, l'accesso remoto alle istanze tramite AWS Systems Manager Session Manager e Run Command è crittografato utilizzando TLS 1.2 e le richieste per creare una connessione vengono firmate utilizzando SigV4. Per informazioni sulla connessione a un'istanza Amazon GameLift Servers gestita, consulta [Connect a flotte di istanze](#).

Gestione delle identità e degli accessi per l' Amazon GameLift Servers

AWS Identity and Access Management (IAM) è un programma Servizio AWS che aiuta un amministratore a controllare in modo sicuro l'accesso alle AWS risorse. Gli amministratori IAM controllano chi è autenticato (accesso effettuato) e autorizzato (dispone di autorizzazioni) a utilizzare risorse Amazon GameLift Servers. IAM è un software Servizio AWS che puoi utilizzare senza costi aggiuntivi.

Argomenti

- [Destinatari](#)
- [Autenticazione con identità](#)
- [Gestione dell'accesso tramite policy](#)
- [Funzionamento di Amazon GameLift Servers con IAM](#)

- [Esempi di policy basate su identità per Amazon GameLift Servers](#)
- [Risoluzione dei problemi di identità e accesso in Amazon GameLift Servers](#)
- [AWS politiche gestite per Amazon GameLift Servers](#)

Destinatari

Il modo in cui utilizzi AWS Identity and Access Management (IAM) varia in base al tuo ruolo:

- Utente del servizio: richiedi le autorizzazioni all'amministratore se non riesci ad accedere alle funzionalità (consulta [Risoluzione dei problemi di identità e accesso in Amazon GameLift Servers](#))
- Amministratore del servizio: determina l'accesso degli utenti e invia le richieste di autorizzazione (consulta [Funzionamento di Amazon GameLift Servers con IAM](#))
- Amministratore IAM: scrivi policy per gestire l'accesso (consulta [Esempi di policy basate su identità per Amazon GameLift Servers](#))

Autenticazione con identità

L'autenticazione è il modo in cui accedi AWS utilizzando le tue credenziali di identità. Devi autenticarti come utente IAM o assumendo un ruolo IAM. Utente root dell'account AWS

Puoi accedere come identità federata utilizzando credenziali provenienti da una fonte di identità come AWS IAM Identity Center (IAM Identity Center), autenticazione Single Sign-On o credenziali. Google/Facebook Per ulteriori informazioni sull'accesso, consulta [Come accedere all' Account AWS](#) nella Guida per l'utente di Accedi ad AWS .

Per l'accesso programmatico, AWS fornisce un SDK e una CLI per firmare crittograficamente le richieste. Per ulteriori informazioni, consulta [AWS Signature Version 4 per le richieste API](#) nella Guida per l'utente di IAM.

Account AWS utente root

Quando si crea un Account AWS, si inizia con un'identità di accesso denominata utente Account AWS root che ha accesso completo a tutte Servizi AWS le risorse. Consigliamo vivamente di non utilizzare l'utente root per le attività quotidiane. Per le attività che richiedono le credenziali dell'utente root, consulta [Attività che richiedono le credenziali dell'utente root](#) nella Guida per l'utente IAM.

Identità federata

Come procedura ottimale, richiedi agli utenti umani di utilizzare la federazione con un provider di identità per accedere Servizi AWS utilizzando credenziali temporanee.

Un'identità federata è un utente della directory aziendale, del provider di identità Web o Directory Service che accede Servizi AWS utilizzando le credenziali di una fonte di identità. Le identità federate assumono ruoli che forniscono credenziali temporanee.

Per la gestione centralizzata degli accessi, si consiglia di utilizzare AWS IAM Identity Center. Per ulteriori informazioni, consulta [Che cos'è il Centro identità IAM?](#) nella Guida per l'utente di AWS IAM Identity Center .

Utenti e gruppi IAM

Un [utente IAM](#) è una identità che dispone di autorizzazioni specifiche per una singola persona o applicazione. Ti consigliamo di utilizzare credenziali temporanee invece di utenti IAM con credenziali a lungo termine. Per ulteriori informazioni, consulta [Richiedere agli utenti umani di utilizzare la federazione con un provider di identità per accedere AWS utilizzando credenziali temporanee](#) nella Guida per l'utente IAM.

Un [gruppo IAM](#) specifica una raccolta di utenti IAM e semplifica la gestione delle autorizzazioni per gestire gruppi di utenti di grandi dimensioni. Per ulteriori informazioni, consulta [Casi d'uso per utenti IAM](#) nella Guida per l'utente di IAM.

Ruoli IAM

Un [ruolo IAM](#) è un'identità con autorizzazioni specifiche che fornisce credenziali temporanee. Puoi assumere un ruolo [passando da un ruolo utente a un ruolo IAM \(console\)](#) o chiamando un'operazione AWS CLI o AWS API. Per ulteriori informazioni, consulta [Metodi per assumere un ruolo](#) nella Guida per l'utente di IAM.

I ruoli IAM sono utili per l'accesso degli utenti federati, le autorizzazioni utente IAM temporanee, l'accesso multi-account, l'accesso multi-servizio e le applicazioni in esecuzione su Amazon EC2. Per maggiori informazioni, consultare [Accesso a risorse multi-account in IAM](#) nella Guida per l'utente IAM.

Gestione dell'accesso tramite policy

Puoi controllare l'accesso AWS creando policy e associandole a AWS identità o risorse. Una policy definisce le autorizzazioni quando è associata a un'identità o a una risorsa. AWS valuta queste

politiche quando un preside effettua una richiesta. La maggior parte delle politiche viene archiviata AWS come documenti JSON. Per maggiori informazioni sui documenti delle policy JSON, consulta [Panoramica delle policy JSON](#) nella Guida per l'utente IAM.

Utilizzando le policy, gli amministratori specificano chi ha accesso a cosa definendo quale principale può eseguire azioni su quali risorse e in quali condizioni.

Per impostazione predefinita, utenti e ruoli non dispongono di autorizzazioni. Un amministratore IAM crea le policy IAM e le aggiunge ai ruoli, che gli utenti possono quindi assumere. Le policy IAM definiscono le autorizzazioni indipendentemente dal metodo utilizzato per eseguirle.

Policy basate sull'identità

Le policy basate su identità sono documenti di policy di autorizzazione JSON che è possibile collegare a un'identità (utente, gruppo o ruolo). Tali policy controllano le operazioni autorizzate per l'identità, nonché le risorse e le condizioni in cui possono essere eseguite. Per informazioni su come creare una policy basata su identità, consultare [Definizione di autorizzazioni personalizzate IAM con policy gestite dal cliente](#) nella Guida per l'utente IAM.

Le policy basate su identità possono essere policy in linea (con embedding direttamente in una singola identità) o policy gestite (policy autonome collegate a più identità). Per informazioni su come scegliere tra una policy gestita o una policy inline, consulta [Scegliere tra policy gestite e policy in linea](#) nella Guida per l'utente di IAM.

Policy basate sulle risorse

Le policy basate su risorse sono documenti di policy JSON che è possibile collegare a una risorsa. Gli esempi includono le policy di trust dei ruoli IAM e le policy dei bucket di Amazon S3. Nei servizi che supportano policy basate sulle risorse, gli amministratori dei servizi possono utilizzarli per controllare l'accesso a una risorsa specifica. In una policy basata sulle risorse è obbligatorio [specificare un'entità principale](#).

Le policy basate sulle risorse sono policy inline che si trovano in tale servizio. Non è possibile utilizzare le policy AWS gestite di IAM in una policy basata sulle risorse.

Altri tipi di policy

AWS supporta tipi di policy aggiuntivi che possono impostare le autorizzazioni massime concesse dai tipi di policy più comuni:

- Limiti delle autorizzazioni: imposta il numero massimo di autorizzazioni che una policy basata su identità ha la possibilità di concedere a un'entità IAM. Per ulteriori informazioni, consulta [Limiti delle autorizzazioni per le entità IAM](#) nella Guida per l'utente di IAM.
- Politiche di controllo del servizio (SCPs): specificano le autorizzazioni massime per un'organizzazione o un'unità organizzativa in AWS Organizations. Per ulteriori informazioni, consultare [Policy di controllo dei servizi](#) nella Guida per l'utente di AWS Organizations.
- Politiche di controllo delle risorse (RCPs): imposta le autorizzazioni massime disponibili per le risorse nei tuoi account. Per ulteriori informazioni, consulta [Politiche di controllo delle risorse \(RCPs\)](#) nella Guida per l'AWS Organizations utente.
- Policy di sessione: policy avanzate passate come parametro quando si crea una sessione temporanea per un ruolo o un utente federato. Per maggiori informazioni, consultare [Policy di sessione](#) nella Guida per l'utente IAM.

Più tipi di policy

Quando a una richiesta si applicano più tipi di policy, le autorizzazioni risultanti sono più complicate da comprendere. Per scoprire come si AWS determina se consentire o meno una richiesta quando sono coinvolti più tipi di policy, consulta [Logica di valutazione delle policy](#) nella IAM User Guide.

Funzionamento di Amazon GameLift Servers con IAM

Prima di utilizzare IAM per gestire l'accesso a Amazon GameLift Servers, scopri quali funzionalità di IAM sono disponibili per l'uso con Amazon GameLift Servers.

Funzionalità IAM che è possibile utilizzare con Amazon GameLift Servers

Funzionalità IAM	Supporto di Amazon GameLift Servers
Policy basate sull'identità	Sì
Policy basate su risorse	No
Operazioni di policy	Sì
Risorse relative alle policy	Sì
Chiavi di condizione della policy (specifica del servizio)	Sì

Funzionalità IAM	Supporto di Amazon GameLift Servers
ACLs	No
ABAC (tag nelle policy)	Sì
Credenziali temporanee	Sì
Autorizzazioni del principale	Sì
Ruoli di servizio	Sì
Ruoli collegati al servizio	No

Per avere una visione di alto livello di come Amazon GameLift Servers e altri AWS servizi funzionano con la maggior parte delle funzionalità IAM, consulta [AWS i servizi che funzionano con IAM nella IAM User Guide](#).

Policy basate su identità per Amazon GameLift Servers

Supporta le policy basate sull'identità: sì

Le policy basate sull'identità sono documenti di policy di autorizzazione JSON che è possibile allegare a un'identità (utente, gruppo di utenti o ruolo IAM). Tali policy definiscono le operazioni che utenti e ruoli possono eseguire, su quali risorse e in quali condizioni. Per informazioni su come creare una policy basata su identità, consulta [Definizione di autorizzazioni personalizzate IAM con policy gestite dal cliente](#) nella Guida per l'utente di IAM.

Con le policy basate sull'identità di IAM, è possibile specificare quali operazioni e risorse sono consentite o respinte, nonché le condizioni in base alle quali le operazioni sono consentite o respinte. Per informazioni su tutti gli elementi utilizzabili in una policy JSON, consulta [Guida di riferimento agli elementi delle policy JSON IAM](#) nella Guida per l'utente IAM.

Esempi di policy basate su identità per Amazon GameLift Servers

Per visualizzare esempi di policy basate su identità Amazon GameLift Servers, consulta [Esempi di policy basate su identità per Amazon GameLift Servers](#).

Policy basate su risorse all'interno di Amazon GameLift Servers

Supporta le policy basate su risorse: no

Le policy basate su risorse sono documenti di policy JSON che è possibile collegare a una risorsa. Esempi di policy basate sulle risorse sono le policy di attendibilità dei ruoli IAM e le policy di bucket Amazon S3. Nei servizi che supportano policy basate sulle risorse, gli amministratori dei servizi possono utilizzarli per controllare l'accesso a una risorsa specifica. Quando è collegata a una risorsa, una policy definisce le operazioni che un principale può eseguire su tale risorsa e a quali condizioni. In una policy basata sulle risorse è obbligatorio [specificare un'entità principale](#). I principali possono includere account, utenti, ruoli, utenti federati o. Servizi AWS

Per consentire l'accesso multi-account, è possibile specificare un intero account o entità IAM in un altro account come entità principale in una policy basata sulle risorse. Per ulteriori informazioni, consulta [Accesso a risorse multi-account in IAM](#) nella Guida per l'utente IAM.

Operazioni di policy per Amazon GameLift Servers

Supporta le operazioni di policy: si

Gli amministratori possono utilizzare le policy AWS JSON per specificare chi ha accesso a cosa. In altre parole, quale entità principale può eseguire operazioni su quali risorse e in quali condizioni.

L'elemento `Action` di una policy JSON descrive le operazioni che è possibile utilizzare per consentire o negare l'accesso in una policy. Includere le operazioni in una policy per concedere le autorizzazioni a eseguire l'operazione associata.

Per un elenco di Amazon GameLift Servers azioni, vedere [Azioni definite da Amazon GameLift Servers](#) nel Service Authorization Reference.

Le operazioni delle policy in Amazon GameLift Servers utilizzano il seguente prefisso prima dell'operazione:

```
gamelift
```

Per specificare più operazioni in una sola istruzione, occorre separarle con la virgola.

```
"Action": [  
    "gamelift:action1",
```

```
"gamelift:action2"  
]
```

È possibile specificare più azioni tramite caratteri jolly (*). Ad esempio, per specificare tutte le azioni che iniziano con la parola `Describe`, includi la seguente azione:

```
"Action": "gamelift:Describe*"
```

Per visualizzare esempi di policy basate su identità Amazon GameLift Servers, consulta [Esempi di policy basate su identità per Amazon GameLift Servers](#).

Risorse relative alle policy per Amazon GameLift Servers

Supporta le risorse relative alle policy: sì

Gli amministratori possono utilizzare le policy AWS JSON per specificare chi ha accesso a cosa. In altre parole, quale entità principale può eseguire operazioni su quali risorse e in quali condizioni.

L'elemento JSON `Resource` della policy specifica l'oggetto o gli oggetti ai quali si applica l'operazione. Come best practice, specifica una risorsa utilizzando il suo [nome della risorsa Amazon \(ARN\)](#). Per le azioni che non supportano le autorizzazioni a livello di risorsa, si utilizza un carattere jolly (*) per indicare che l'istruzione si applica a tutte le risorse.

```
"Resource": "*"
```

Per un elenco dei tipi di Amazon GameLift Servers risorse e relativi ARNs, vedere [Resources defined by Amazon GameLift Servers](#) nel Service Authorization Reference. Per informazioni sulle operazioni con cui è possibile specificare l'ARN di ogni risorsa, consulta la sezione [Operazioni definite da Amazon GameLift Servers](#).

Alcune Amazon GameLift Servers risorse hanno valori ARN, che consentono di gestire l'accesso alle risorse utilizzando le policy IAM. La risorsa parco istanze di Amazon GameLift Servers ha un ARN con la seguente sintassi:

```
arn:${Partition}:gamelift:${Region}:${Account}:fleet/${FleetId}
```

Per ulteriori informazioni sul formato di ARNs, consulta [Amazon Resource Names \(ARNs\)](#) nel Riferimenti generali di AWS.

Ad esempio, per specificare il parco istanze `fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa` nell'istruzione, utilizza il seguente ARN:

```
"Resource": "arn:aws:gamelift:us-west-2:123456789012:fleet/fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa"
```

Per specificare tutte le flotte che appartengono a un account specifico, usa un carattere jolly (*):

```
"Resource": "arn:aws:gamelift:us-west-2:123456789012:fleet/*"
```

Per visualizzare esempi di policy basate su identità Amazon GameLift Servers, consulta [Esempi di policy basate su identità per Amazon GameLift Servers](#).

Chiavi di condizione delle policy per Amazon GameLift Servers

Supporta le chiavi di condizione delle policy specifiche del servizio: sì

Gli amministratori possono utilizzare le policy AWS JSON per specificare chi ha accesso a cosa. In altre parole, quale entità principale può eseguire operazioni su quali risorse e in quali condizioni.

L'elemento `Condition` specifica quando le istruzioni vengono eseguite in base a criteri definiti. È possibile compilare espressioni condizionali che utilizzano [operatori di condizione](#), ad esempio uguale a o minore di, per soddisfare la condizione nella policy con i valori nella richiesta. Per visualizzare tutte le chiavi di condizione AWS globali, consulta le chiavi di [contesto delle condizioni AWS globali nella Guida](#) per l'utente IAM.

Per un elenco delle chiavi di Amazon GameLift Servers condizione, consulta [Condition keys for Amazon GameLift Servers](#) nel Service Authorization Reference. Per informazioni su operazioni e risorse con cui è possibile utilizzare una chiave di condizione, consulta la sezione [Operazioni definite da Amazon GameLift Servers](#).

Per visualizzare esempi di policy basate su identità Amazon GameLift Servers, consulta [Esempi di policy basate su identità per Amazon GameLift Servers](#).

ACLs in Amazon GameLift Servers

Supporti ACLs: no

Le liste di controllo degli accessi (ACLs) controllano quali principali (membri dell'account, utenti o ruoli) dispongono delle autorizzazioni per accedere a una risorsa. ACLs sono simili alle politiche basate sulle risorse, sebbene non utilizzino il formato del documento di policy JSON.

ABAC con Amazon GameLift Servers

Supporta ABAC (tag nelle policy): sì

Il controllo degli accessi basato su attributi (ABAC) è una strategia di autorizzazione che definisce le autorizzazioni in base ad attributi chiamati tag. È possibile allegare tag a entità e AWS risorse IAM, quindi progettare politiche ABAC per consentire operazioni quando il tag del principale corrisponde al tag sulla risorsa.

Per controllare l'accesso basato su tag, fornire informazioni sui tag nell'[elemento condizione](#) di una policy utilizzando le chiavi di condizione `aws:ResourceTag/key-name`, `aws:RequestTag/key-name` o `aws:TagKeys`.

Se un servizio supporta tutte e tre le chiavi di condizione per ogni tipo di risorsa, il valore per il servizio è Sì. Se un servizio supporta tutte e tre le chiavi di condizione solo per alcuni tipi di risorsa, allora il valore sarà Parziale.

Per maggiori informazioni su ABAC, consulta [Definizione delle autorizzazioni con autorizzazione ABAC](#) nella Guida per l'utente di IAM. Per visualizzare un tutorial con i passaggi per l'impostazione di ABAC, consulta [Utilizzo del controllo degli accessi basato su attributi \(ABAC\)](#) nella Guida per l'utente di IAM.

Per un esempio di politica basata sull'identità che limita l'accesso a una risorsa in base ai tag presenti su quella risorsa, vedi. [Visualizzare i parchi istanze Amazon GameLift Servers in base ai tag](#)

Utilizzo di credenziali temporanee con Amazon GameLift Servers

Supporta le credenziali temporanee: sì

Le credenziali temporanee forniscono l'accesso a breve termine alle AWS risorse e vengono create automaticamente quando si utilizza la federazione o si cambia ruolo. AWS consiglia di generare dinamicamente credenziali temporanee anziché utilizzare chiavi di accesso a lungo termine. Per ulteriori informazioni, consulta [Credenziali di sicurezza temporanee in IAM](#) e [Servizi AWS compatibili con IAM](#) nelle Guida per l'utente IAM.

Autorizzazioni del principale tra servizi per Amazon GameLift Servers

Supporta l'inoltro delle sessioni di accesso (FAS): sì

Le sessioni di accesso inoltrato (FAS) utilizzano le autorizzazioni del principale che chiama e, in combinazione con la richiesta Servizio AWS, Servizio AWS per effettuare richieste ai servizi downstream. Per i dettagli delle policy relative alle richieste FAS, consulta la pagina [Inoltrare sessioni di accesso](#).

Ruoli di servizio per Amazon GameLift Servers

Supporta i ruoli di servizio: sì

Un ruolo di servizio è un [ruolo IAM](#) che un servizio assume per eseguire operazioni per tuo conto. Un amministratore IAM può creare, modificare ed eliminare un ruolo di servizio dall'interno di IAM. Per ulteriori informazioni, consulta la sezione [Creazione di un ruolo per delegare le autorizzazioni a un Servizio AWS](#) nella Guida per l'utente di IAM.

Warning

La modifica delle autorizzazioni per un ruolo di servizio potrebbe compromettere la funzionalità di Amazon GameLift Servers. Modifica i ruoli di servizio solo quando Amazon GameLift Servers fornisce le indicazioni per farlo.

Consenti ai server Amazon GameLift Servers di gioco ospitati di accedere ad altre AWS risorse, come una AWS Lambda funzione o un database Amazon DynamoDB. Poiché i server di gioco sono ospitati su flotte Amazon GameLift Servers gestite, è necessario un ruolo di servizio che consenta un accesso Amazon GameLift Servers limitato alle altre risorse. AWS Per ulteriori informazioni, consulta [Connect il server di gioco Amazon GameLift Servers ospitato ad altre AWS risorse](#).

Ruoli collegati ai servizi per l'Amazon GameLift Servers

Supporta i ruoli collegati ai servizi: no

Un ruolo collegato al servizio è un tipo di ruolo di servizio collegato a un. Servizio AWS Il servizio può assumere il ruolo per eseguire un'operazione per tuo conto. I ruoli collegati al servizio vengono visualizzati nel tuo account Account AWS e sono di proprietà del servizio. Un amministratore IAM può visualizzare le autorizzazioni per i ruoli collegati al servizio, ma non modificarle.

Per i dettagli sulla creazione o la gestione di ruoli collegati ai servizi, consulta i [AWS servizi che funzionano con IAM nella IAM](#) User Guide. Trova un servizio nella tabella che include un Yes nella colonna Ruoli collegati ai servizi. Scegli Sì per visualizzare la documentazione del ruolo collegato al servizio per quel servizio.

Esempi di policy basate su identità per Amazon GameLift Servers

Per impostazione predefinita, gli utenti e i ruoli non dispongono dell'autorizzazione per creare o modificare risorse Amazon GameLift Servers. Per concedere agli utenti l'autorizzazione a eseguire azioni sulle risorse di cui hanno bisogno, un amministratore IAM può creare policy IAM.

Per informazioni su come creare una policy basata su identità IAM utilizzando questi documenti di policy JSON di esempio, consulta [Creazione di policy IAM \(console\)](#) nella Guida per l'utente di IAM.

Per informazioni dettagliate sulle azioni e sui tipi di risorse definiti da Amazon GameLift Servers, incluso il formato di ARNs per ogni tipo di risorsa, consulta [Azioni, risorse e chiavi di condizione Amazon GameLift Servers](#) nel Service Authorization Reference.

Argomenti

- [Best practice per le policy](#)
- [Utilizzo della console di Amazon GameLift Servers](#)
- [Consentire agli utenti di visualizzare le loro autorizzazioni](#)
- [Consentire l'accesso ai giocatori per le sessioni di gioco](#)
- [Consenti l'accesso a una Amazon GameLift Servers coda](#)
- [Visualizzare i parchi istanze Amazon GameLift Servers in base ai tag](#)
- [Accedi a un file di build del gioco in Amazon S3](#)

Best practice per le policy

Le policy basate su identità determinano se qualcuno può creare, accedere o eliminare risorse Amazon GameLift Servers nel tuo account. Queste operazioni possono comportare costi aggiuntivi per l'Account AWS. Quando si creano o modificano policy basate sull'identità, seguire queste linee guida e raccomandazioni:

- Inizia con le policy AWS gestite e passa alle autorizzazioni con privilegi minimi: per iniziare a concedere autorizzazioni a utenti e carichi di lavoro, utilizza le politiche AWS gestite che concedono le autorizzazioni per molti casi d'uso comuni. Sono disponibili nel tuo Account AWS. Ti consigliamo di ridurre ulteriormente le autorizzazioni definendo politiche gestite dai clienti specifiche per i tuoi casi d'uso. Per maggiori informazioni, consulta [Policy gestite da AWS](#) o [Policy gestite da AWS per le funzioni dei processi](#) nella Guida per l'utente di IAM.
- Applicazione delle autorizzazioni con privilegio minimo - Quando si impostano le autorizzazioni con le policy IAM, concedere solo le autorizzazioni richieste per eseguire un'attività. È possibile farlo

definendo le azioni che possono essere intraprese su risorse specifiche in condizioni specifiche, note anche come autorizzazioni con privilegio minimo. Per maggiori informazioni sull'utilizzo di IAM per applicare le autorizzazioni, consulta [Policy e autorizzazioni in IAM](#) nella Guida per l'utente di IAM.

- Condizioni d'uso nelle policy IAM per limitare ulteriormente l'accesso - Per limitare l'accesso ad azioni e risorse è possibile aggiungere una condizione alle policy. Ad esempio, è possibile scrivere una condizione di policy per specificare che tutte le richieste devono essere inviate utilizzando SSL. Puoi anche utilizzare le condizioni per concedere l'accesso alle azioni del servizio se vengono utilizzate tramite uno specifico Servizio AWS, ad esempio CloudFormation. Per maggiori informazioni, consultare la sezione [Elementi delle policy JSON di IAM: condizione](#) nella Guida per l'utente di IAM.
- Utilizzo dello strumento di analisi degli accessi IAM per convalidare le policy IAM e garantire autorizzazioni sicure e funzionali - Lo strumento di analisi degli accessi IAM convalida le policy nuove ed esistenti in modo che aderiscano al linguaggio (JSON) della policy IAM e alle best practice di IAM. Lo strumento di analisi degli accessi IAM offre oltre 100 controlli delle policy e consigli utili per creare policy sicure e funzionali. Per maggiori informazioni, consultare [Convalida delle policy per il Sistema di analisi degli accessi IAM](#) nella Guida per l'utente di IAM.
- Richiedi l'autenticazione a più fattori (MFA): se hai uno scenario che richiede utenti IAM o un utente root nel Account AWS tuo, attiva l'MFA per una maggiore sicurezza. Per richiedere la MFA quando vengono chiamate le operazioni API, aggiungere le condizioni MFA alle policy. Per maggiori informazioni, consultare [Protezione dell'accesso API con MFA](#) nella Guida per l'utente di IAM.

Per maggiori informazioni sulle best practice in IAM, consultare [Best practice di sicurezza in IAM](#) nella Guida per l'utente IAM.

Utilizzo della console di Amazon GameLift Servers

Per accedere alla console Amazon GameLift Servers è necessario disporre di un insieme di autorizzazioni minimo. Queste autorizzazioni devono consentirti di elencare e visualizzare i dettagli sulle risorse del Amazon GameLift Servers tuo. Account AWS Se crei una policy basata sull'identità più restrittiva rispetto alle autorizzazioni minime richieste, la console non funzionerà nel modo previsto per le entità (utenti o ruoli) associate a tale policy.

Per garantire che tali entità possano ancora utilizzare la Amazon GameLift Servers console, aggiungi le autorizzazioni a utenti e gruppi con la sintassi riportata negli esempi seguenti e versioni successive. [Esempi di autorizzazioni amministrative](#) Per ulteriori informazioni, consulta [Imposta le autorizzazioni utente per Amazon GameLift Servers](#).

Gli utenti che utilizzano operazioni Amazon GameLift Servers tramite AWS CLI o AWS API non richiedono autorizzazioni minime per la console. Puoi invece limitare l'accesso solo alle operazioni che l'utente deve eseguire. Ad esempio, un utente giocatore, che agisce per conto dei client di gioco, richiede l'accesso per richiedere sessioni di gioco, inserire giocatori nei giochi e altre attività.

Per informazioni sulle autorizzazioni necessarie per utilizzare tutte le funzionalità della Amazon GameLift Servers console, consulta la sintassi delle autorizzazioni per gli amministratori in. [Esempi di autorizzazioni amministrative](#)

Consentire agli utenti di visualizzare le loro autorizzazioni

Questo esempio mostra in che modo è possibile creare una policy che consente agli utenti IAM di visualizzare le policy inline e gestite che sono collegate alla relativa identità utente. Questa politica include le autorizzazioni per completare questa azione sulla console o utilizzando l'API o a livello di codice. AWS CLI AWS

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ViewOwnUserInfo",
      "Effect": "Allow",
      "Action": [
        "iam:GetUserPolicy",
        "iam:ListGroupsForUser",
        "iam:ListAttachedUserPolicies",
        "iam:ListUserPolicies",
        "iam:GetUser"
      ],
      "Resource": ["arn:aws:iam::*:user/${aws:username}"]
    },
    {
      "Sid": "NavigateInConsole",
      "Effect": "Allow",
      "Action": [
        "iam:GetGroupPolicy",
        "iam:GetPolicyVersion",
        "iam:GetPolicy",
        "iam:ListAttachedGroupPolicies",
        "iam:ListGroupPolicies",
        "iam:ListPolicyVersions",
        "iam:ListPolicies",

```

```

        "iam:ListUsers"
      ],
      "Resource": "*"
    }
  ]
}

```

Consentire l'accesso ai giocatori per le sessioni di gioco

Per inserire i giocatori nelle sessioni di gioco, i client di gioco e i servizi di backend necessitano di autorizzazioni. Per esempi di policy per questi scenari, consulta. [Esempi di autorizzazioni utente per i giocatori](#)

Consenti l'accesso a una Amazon GameLift Servers coda

L'esempio seguente fornisce a un utente l'accesso a una Amazon GameLift Servers coda specifica.

Questa politica concede all'utente le autorizzazioni per aggiungere, aggiornare ed eliminare le destinazioni della coda con le seguenti azioni: `gamelift:UpdateGameSessionQueue`, e. `gamelift>DeleteGameSessionQueue` `gamelift:DescribeGameSessionQueues` Come illustrato, questo criterio utilizza l'`Resource` elemento per limitare l'accesso a una singola coda: `gamesessionqueue/examplequeue123`

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ViewSpecificQueueInfo",
      "Effect": "Allow",
      "Action": [
        "gamelift:DescribeGameSessionQueues"
      ],
      "Resource": "arn:aws:gamelift:us-east-1:555555555555:gamesessionqueue/examplequeue123"
    },
    {
      "Sid": "ManageSpecificQueue",
      "Effect": "Allow",
      "Action": [

```

```

        "gamelift:UpdateGameSessionQueue",
        "gamelift>DeleteGameSessionQueue"
    ],
    "Resource": "arn:aws:gamelift:us-east-1:111122223333:gamesessionqueue/
examplequeue123"
    }
]
}

```

Visualizzare i parchi istanze Amazon GameLift Servers in base ai tag

È possibile utilizzare le condizioni nella policy basata sulle identità per controllare l'accesso alle risorse di Amazon GameLift Servers in base ai tag. Questo esempio mostra come è possibile creare una politica che consenta di visualizzare una flotta se il Owner tag corrisponde al nome utente dell'utente. Questa politica concede anche le autorizzazioni necessarie per completare questa operazione nella console.

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ListFleetsInConsole",
      "Effect": "Allow",
      "Action": "gamelift:ListFleets",
      "Resource": "*"
    },
    {
      "Sid": "ViewFleetIfOwner",
      "Effect": "Allow",
      "Action": "gamelift:DescribeFleetAttributes",
      "Resource": "arn:aws:gamelift:*:*:fleet/*",
      "Condition": {
        "StringEquals": {"aws:ResourceAccount": "${aws:username}"}
      }
    }
  ]
}

```

Accedi a un file di build del gioco in Amazon S3

Dopo aver integrato il server di gioco con Amazon GameLift Servers, carica i file di build su Amazon S3. Per accedere Amazon GameLift Servers ai file di build, utilizza la seguente politica.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion"
      ],
      "Resource": "arn:aws:s3:::bucket-name/object-name"
    }
  ]
}
```

Per ulteriori informazioni sul caricamento dei file di Amazon GameLift Servers gioco, consulta [Crea un server di gioco costruito per Amazon GameLift Servers](#).

Risoluzione dei problemi di identità e accesso in Amazon GameLift Servers

Utilizza le seguenti informazioni per aiutarti a diagnosticare e risolvere i problemi più comuni che potresti riscontrare quando lavori con Amazon GameLift Servers and AWS Identity and Access Management (IAM).

Argomenti

- [Non sono autorizzato a eseguire un'operazione in Amazon GameLift Servers](#)
- [Non sono autorizzato a eseguire iam: PassRole](#)
- [Voglio consentire a persone esterne a me di accedere Account AWS alle mie Amazon GameLift Servers risorse](#)

Non sono autorizzato a eseguire un'operazione in Amazon GameLift Servers

Se ti Console di gestione AWS dice che non sei autorizzato a eseguire un'azione, contatta l'amministratore del tuo AWS account per ricevere assistenza. L'amministratore è colui che ti ha fornito le credenziali di accesso.

Il seguente errore di esempio si verifica quando l'utente `mateojackson` IAM tenta di utilizzare la console per visualizzare i dettagli su una coda ma non dispone delle `gamelift:DescribeGameSessionQueues` autorizzazioni:

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
gamelift:DescribeGameSessionQueues on resource: examplequeue123
```

In questo caso, Mateo chiede al suo amministratore di aggiornare le sue politiche per consentirgli l'accesso in lettura alla `examplequeue123` risorsa utilizzando l'azione `gamelift:DescribeGameSessionQueues`.

Non sono autorizzato a eseguire `iam:PassRole`

Se ricevi un errore che indica che non sei autorizzato a eseguire l'operazione `iam:PassRole`, le tue policy devono essere aggiornate per poter passare un ruolo a Amazon GameLift Servers.

Alcuni Servizi AWS consentono di passare un ruolo esistente a quel servizio invece di creare un nuovo ruolo di servizio o un ruolo collegato al servizio. Per eseguire questa operazione, è necessario disporre delle autorizzazioni per trasmettere il ruolo al servizio.

L'errore di esempio seguente si verifica quando un utente IAM denominato `marymajor` cerca di utilizzare la console per eseguire un'operazione in Amazon GameLift Servers. Tuttavia, l'operazione richiede che il servizio disponga delle autorizzazioni concesse da un ruolo di servizio. Mary non dispone delle autorizzazioni per trasmettere il ruolo al servizio.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
iam:PassRole
```

In questo caso, le policy di Mary devono essere aggiornate per poter eseguire l'operazione `iam:PassRole`.

Se hai bisogno di aiuto, contatta il tuo AWS amministratore. L'amministratore è la persona che ti ha fornito le credenziali di accesso.

Voglio consentire a persone esterne a me di accedere Account AWS alle mie Amazon GameLift Servers risorse

È possibile creare un ruolo con il quale utenti in altri account o persone esterne all'organizzazione possono accedere alle tue risorse. È possibile specificare chi è attendibile per l'assunzione del ruolo. Per i servizi che supportano politiche basate sulle risorse o liste di controllo degli accessi (ACLs), puoi utilizzare tali politiche per concedere alle persone l'accesso alle tue risorse.

Per maggiori informazioni, consulta gli argomenti seguenti:

- Per capire se Amazon GameLift Servers supporta queste funzionalità, consulta [Funzionamento di Amazon GameLift Servers con IAM](#).
- Per scoprire come fornire l'accesso alle tue risorse attraverso Account AWS le risorse di tua proprietà, consulta [Fornire l'accesso a un utente IAM in un altro Account AWS di tua proprietà](#) nella IAM User Guide.
- Per scoprire come fornire l'accesso alle tue risorse a terze parti Account AWS, consulta [Fornire l'accesso a soggetti Account AWS di proprietà di terze parti](#) nella Guida per l'utente IAM.
- Per informazioni su come fornire l'accesso tramite la federazione delle identità, consulta [Fornire l'accesso a utenti autenticati esternamente \(federazione delle identità\)](#) nella Guida per l'utente IAM.
- Per informazioni sulle differenze di utilizzo tra ruoli e policy basate su risorse per l'accesso multi-account, consulta [Accesso a risorse multi-account in IAM](#) nella Guida per l'utente IAM.

AWS politiche gestite per Amazon GameLift Servers

Una politica AWS gestita è una politica autonoma creata e amministrata da AWS. AWS le politiche gestite sono progettate per fornire autorizzazioni per molti casi d'uso comuni, in modo da poter iniziare ad assegnare autorizzazioni a utenti, gruppi e ruoli.

Tieni presente che le policy AWS gestite potrebbero non concedere le autorizzazioni con il privilegio minimo per i tuoi casi d'uso specifici, poiché sono disponibili per tutti i clienti. AWS Si consiglia pertanto di ridurre ulteriormente le autorizzazioni definendo [policy gestite dal cliente](#) specifiche per i propri casi d'uso.

Non è possibile modificare le autorizzazioni definite nelle politiche gestite. AWS Se AWS aggiorna le autorizzazioni definite in una politica AWS gestita, l'aggiornamento ha effetto su tutte le identità principali (utenti, gruppi e ruoli) a cui è associata la politica. AWS è più probabile che aggiorni una

policy AWS gestita quando ne Servizio AWS viene lanciata una nuova o quando diventano disponibili nuove operazioni API per i servizi esistenti.

Per ulteriori informazioni, consultare [Policy gestite da AWS](#) nella Guida per l'utente di IAM.

AWS politica gestita: GameLiftContainerFleetPolicy

Puoi collegarti GameLiftContainerFleetPolicy ai tuoi ruoli IAM.

La policy concede le autorizzazioni per le azioni di calcolo in una Amazon GameLift Servers flotta di container. Una flotta di container è un insieme di risorse di hosting che Amazon GameLift Servers gestisce per te. Amazon GameLift Servers necessita delle autorizzazioni per connettersi al Amazon GameLift Servers servizio e ad altri AWS servizi per tuo conto.

Quando crei una flotta di container con Amazon GameLift Servers, fornisci un ruolo di servizio IAM con la policy GameLiftContainerFleetPolicy gestita allegata. Per istruzioni sulla creazione del ruolo di servizio, consulta [Configurare un ruolo di servizio IAM per Amazon GameLift Servers](#).

Per ulteriori informazioni, consulta [GameLiftContainerFleetPolicy](#).

Dettagli delle autorizzazioni

Questa policy include le seguenti autorizzazioni:

- `cloudwatch`— Consente di Amazon GameLift Servers scrivere i log delle sessioni di gioco in un flusso di log di Amazon CloudWatch Events nel tuo AWS account.
- `cloudwatch` — Consente di Amazon GameLift Servers creare un gruppo di CloudWatch log per organizzare i dati delle sessioni di gioco nel log stream.
- `s3`— Consente di Amazon GameLift Servers scrivere i log delle sessioni di gioco in un bucket Amazon Simple Storage Service nel tuo AWS account.
- `s3`— Consente di Amazon GameLift Servers recuperare la posizione Regione AWS in cui si trova un bucket Amazon S3 specificato, utilizzando l'azione API. `s3:GetBucketLocation`
- `gamelift` — Consente di Amazon GameLift Servers recuperare un token di autenticazione che consente a un server di gioco ospitato di comunicare con il servizio tramite il Amazon GameLift Servers tuo account. AWS

Amazon GameLift Serversaggiornamenti alle politiche AWS gestite

Visualizza i dettagli sugli aggiornamenti delle politiche AWS gestite Amazon GameLift Servers da quando questo servizio ha iniziato a tenere traccia di queste modifiche. Per ricevere avvisi automatici sulle modifiche a questa pagina, iscriviti al feed RSS nella pagina delle [note di Amazon GameLift Servers rilascio](#).

Modifica	Descrizione	Data
GameLiftContainerFleetPolicy — Modifica	Amazon GameLift Serversha aggiunto nuove autorizzazioni per recuperare i dati Regione AWS di un bucket Amazon S3.	5 febbraio 2024
GameLiftContainerFleetPolicy : nuova policy	Amazon GameLift Serversha aggiunto nuove autorizzazioni per consentire l'esecuzione dei contenitori di server di gioco su flotte gestite. Amazon GameLift Servers	12 novembre 2024
Amazon GameLift Servers ha iniziato il rilevamento delle modifiche	Amazon GameLift Serversha iniziato a tenere traccia delle modifiche alle sue politiche AWS gestite.	12 novembre 2024

Registrazione e monitoraggio con Amazon GameLift Servers

Il monitoraggio è una parte importante del mantenimento dell'affidabilità, della disponibilità e delle prestazioni di Amazon GameLift Servers e le tue AWS soluzioni. È necessario raccogliere i dati di monitoraggio da tutte le parti della AWS soluzione in modo da poter eseguire più facilmente il debug di un errore multipunto, se si verifica.

AWS e Amazon GameLift Servers forniscono diversi strumenti per monitorare le risorse di hosting dei giochi e rispondere a potenziali incidenti.

CloudWatch Allarmi Amazon

Utilizzando Amazon CloudWatch alarms, controlli una singola metrica per un periodo di tempo specificato. Se la metrica supera una determinata soglia, viene inviata una notifica a un argomento o a una politica di Auto AWS Scaling di Amazon SNS. CloudWatch gli allarmi vengono attivati quando il loro stato cambia e vengono mantenuti per un determinato numero di periodi, non perché si trovano in uno stato particolare. Per ulteriori informazioni, consulta [Monitora Amazon GameLift Servers con Amazon CloudWatch](#).

AWS CloudTrail Registri

CloudTrail fornisce un registro delle azioni intraprese da un utente, un ruolo o un AWS servizio in Amazon GameLift Servers. Utilizzando le informazioni raccolte da CloudTrail, è possibile determinare la richiesta effettuata a Amazon GameLift Servers, l'indirizzo IP da cui è stata effettuata la richiesta, chi ha effettuato la richiesta, quando è stata effettuata e dettagli aggiuntivi. Per ulteriori informazioni, consulta [Registrazione delle chiamate Amazon GameLift Servers API con AWS CloudTrail](#).

Convalida della conformità per Amazon GameLift Servers

Amazon GameLift Servers non rientra nell'ambito di alcun programma di AWS conformità.

Per sapere se un Servizio AWS programma rientra nell'ambito di specifici programmi di conformità, consulta Servizi AWS la sezione [Scope by Compliance Program Servizi AWS](#) e scegli il programma di conformità che ti interessa. Per informazioni generali, consulta Programmi di [AWS conformità Programmi](#) di di .

È possibile scaricare report di audit di terze parti utilizzando AWS Artifact. Per ulteriori informazioni, consulta [Scaricamento dei report in AWS Artifact](#) .

La vostra responsabilità di conformità durante l'utilizzo Servizi AWS è determinata dalla sensibilità dei dati, dagli obiettivi di conformità dell'azienda e dalle leggi e dai regolamenti applicabili. Per ulteriori informazioni sulla responsabilità di conformità durante l'utilizzo Servizi AWS, consulta [AWS la documentazione sulla sicurezza](#).

Resilienza in Amazon GameLift Servers

Se stai usando Amazon GameLift Servers FleetIQ come funzionalità autonoma di Amazon EC2, consulta la sezione [Security in Amazon EC2 nella Amazon EC2](#) User Guide.

L'infrastruttura AWS globale è costruita attorno a AWS regioni e zone di disponibilità. AWS Le regioni forniscono più zone di disponibilità fisicamente separate e isolate, collegate con reti a bassa latenza, ad alto throughput e altamente ridondanti. Con le zone di disponibilità, puoi progettare e gestire

applicazioni e database che eseguono automaticamente il failover tra zone di disponibilità senza interruzioni. Le zone di disponibilità sono più disponibili, tolleranti ai guasti e scalabili rispetto alle infrastrutture a data center singolo o multiplo tradizionali.

[Per ulteriori informazioni su AWS regioni e zone di disponibilità, consulta infrastruttura globale.AWS](#)

Oltre all'infrastruttura AWS globale, Amazon GameLift Servers offre le seguenti funzionalità per supportare le esigenze di resilienza dei dati:

- **Code multiregionali:** Amazon GameLift Servers le code delle sessioni di gioco vengono utilizzate per inserire nuove sessioni di gioco con le risorse di hosting disponibili. Le code che si estendono su più regioni sono in grado di reindirizzare i posizionamenti delle sessioni di gioco in caso di interruzione regionale. Per ulteriori informazioni e le best practice sulla creazione di code di sessioni di gioco, consulta [Personalizza una coda di sessioni di gioco](#).
- **Scalabilità automatica della capacità:** mantieni l'integrità e la disponibilità delle risorse di hosting utilizzando Amazon GameLift Servers strumenti di ridimensionamento. Questi strumenti offrono una vasta gamma di opzioni che consentono di regolare la capacità del parco istanze in base alle esigenze del gioco e dei giocatori. Per ulteriori informazioni sul dimensionamento, consultare [Scalabilità della capacità di hosting di giochi con Amazon GameLift Servers](#).
- **Distribuzione tra le istanze:** Amazon GameLift Servers distribuisce il traffico in entrata su più istanze, a seconda delle dimensioni del parco veicoli. Come best practice, i giochi in produzione devono avere più istanze per mantenere la disponibilità nel caso in cui un'istanza risulti non integra o non risponde.
- **Storage Amazon S3:** build e script di server di gioco caricati su Amazon GameLift Servers sono archiviati in Amazon S3 utilizzando la classe di storage Standard, che utilizza più repliche di data center per aumentare la resilienza. I log delle sessioni di gioco vengono inoltre archiviati in Amazon S3 utilizzando la classe di storage Standard.

Sicurezza dell'infrastruttura in Amazon GameLift Servers

Se stai usando Amazon GameLift Servers FleetIQ come funzionalità autonoma di Amazon EC2, consulta la sezione [Security in Amazon EC2 nella Amazon EC2](#) User Guide.

Come servizio gestito, Amazon GameLift Servers è protetto dalle procedure di sicurezza della rete AWS globale descritte nel white paper [Amazon Web Services: panoramica dei processi di sicurezza](#).

Utilizzi chiamate API AWS pubblicate per accedere Amazon GameLift Servers attraverso la rete. I client devono supportare Transport Layer Security (TLS) 1.2 o versioni successive. È consigliabile

TLS 1.3 o versioni successive. I client devono, inoltre, supportare le suite di crittografia con PFS (Perfect Forward Secrecy), ad esempio Ephemeral Diffie-Hellman (DHE) o Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). La maggior parte dei sistemi moderni, come Java 7 e versioni successive, supporta tali modalità.

Inoltre, le richieste devono essere firmate utilizzando un ID chiave di accesso e una chiave di accesso segreta associata a un principale IAM. O puoi utilizzare [AWS Security Token Service](#) (AWS STS) per generare credenziali di sicurezza temporanee per sottoscrivere le richieste.

Il Amazon GameLift Servers il servizio colloca tutte le flotte nei cloud privati virtuali di Amazon (VPCs) in modo che ogni flotta esista in un'area logicamente isolata nel AWS cloud. È possibile utilizzare... Amazon GameLift Servers policy per controllare l'accesso da endpoint VPC specifici o specifici. VPCs In effetti, questo isola l'accesso alla rete a un determinato Amazon GameLift Servers risorsa proveniente solo dal VPC specifico all'interno della AWS rete. Quando crei un parco istanze specifici un intervallo di numeri di porta e indirizzi IP. Questi intervalli limitano il modo in cui il traffico in entrata può accedere ai server di gioco ospitati nel VPC di un parco istanze. Utilizza le best practice di sicurezza standard per scegliere le impostazioni di accesso del parco istanze.

Analisi della configurazione e delle vulnerabilità in Amazon GameLift Servers

Se la utilizzi Amazon GameLift Servers FleetIQ come funzionalità autonoma con Amazon EC2, [consulta la sezione Sicurezza in Amazon EC2 nella Guida per l'utente di Amazon EC2](#).

Configurazione e controllo IT sono una responsabilità condivisa tra AWS e te, il nostro cliente. [Per ulteriori informazioni, consulta il modello di responsabilità condivisa.](#) AWS AWS gestisce le attività di sicurezza di base come l'applicazione di patch al sistema operativo guest (OS) e al database, la configurazione del firewall e il disaster recovery. Queste procedure sono state riviste e certificate dalle terze parti appropriate. Per ulteriori dettagli, consulta la seguente risorsa: [Amazon Web Services: panoramica dei processi di sicurezza](#) (white paper).

Le best practice di sicurezza seguenti gestiscono anche l'analisi di configurazione e vulnerabilità in Amazon GameLift Servers:

- I clienti sono responsabili della gestione del software distribuito nelle istanze Amazon GameLift Servers per l'hosting dei giochi. Nello specifico:
 - Il software applicativo del server di gioco fornito dal cliente deve essere gestito, inclusi gli aggiornamenti e le patch di sicurezza. Per aggiornare il software del server di gioco, carica una

nuova build Amazon GameLift Servers o forniscila in una nuova immagine del container, crea una nuova flotta e reindirizza il traffico verso la nuova flotta.

- L'Amazon Machine Image (AMI) di base che include il sistema operativo, viene aggiornata solo quando viene creato un nuovo parco istanze. Per applicare patch, aggiornare e proteggere il sistema operativo e le altre applicazioni che fanno parte dell'AMI, sostituisci le flotte regolarmente, indipendentemente dagli aggiornamenti dei server di gioco. Come minimo, le flotte di produzione devono essere sostituite almeno una volta all'anno. Per ulteriori dettagli, consulta [Mantieni aggiornati gli ambienti di runtime della flotta](#).
- I clienti dovrebbero prendere in considerazione la possibilità di aggiornare regolarmente i propri giochi con le versioni SDK più recenti, tra cui AWS SDK, Amazon GameLift Servers Server SDK e Amazon GameLift Servers Client SDK for Realtime Servers.

Best practice relative alla sicurezza di Amazon GameLift Servers

Se la utilizzi Amazon GameLift Servers FleetIQ come funzionalità autonoma con Amazon EC2, [consulta la sezione Sicurezza in Amazon EC2 nella Guida per l'utente di Amazon EC2](#).

Amazon GameLift Servers fornisce una serie di funzionalità di sicurezza che occorre valutare durante lo sviluppo e l'implementazione delle policy di sicurezza. Le seguenti best practice sono linee guida generali e non rappresentano una soluzione di sicurezza completa. Poiché queste best practice potrebbero non essere appropriate o sufficienti per l'ambiente, sono da considerare come considerazioni utili anziché prescrizioni.

Mantieni aggiornati gli ambienti di runtime della flotta

Amazon GameLift Servers consiglia vivamente di sostituire regolarmente le flotte gestite (comprese le flotte gestite di EC2 e container) per mantenere ambienti di runtime sicuri per i server di gioco. Le flotte che funzionano per periodi prolungati senza aggiornamenti di runtime possono contenere dipendenze obsolete e vulnerabilità di sicurezza che potrebbero compromettere i server di gioco. Per i dettagli su come viene condivisa la responsabilità del software distribuito alle flotte, consulta Amazon GameLift Servers [Analisi della configurazione e delle vulnerabilità in Amazon GameLift Servers](#)

L'ambiente di runtime di una flotta gestita è determinato dalla versione di Amazon Machine Image (AMI). Quando viene creato un nuovo parco macchine, Amazon GameLift Servers assegna al parco l'ultima versione AMI disponibile e tutte le istanze di calcolo di quel parco macchine vengono

distribuite con quella versione. Per aggiornare la versione AMI, devi creare una nuova flotta. Per informazioni dettagliate sulle versioni AMI correnti, vedere [Amazon GameLift Servers Versioni AMI](#).

Pratiche consigliate:

- Monitora l'età della flotta e sostituisci le flotte che hanno più di 30 giorni: puoi tenere traccia della data di creazione di una flotta nella Amazon GameLift Servers console o utilizzare la CLI per recuperare gli attributi della flotta. Amazon GameLift Servers visualizza gli avvisi nella console per le flotte che hanno più di 90 giorni e avvisa i titolari di account via e-mail per le flotte che hanno più di un anno.

 Note

L'aggiornamento di una flotta (ad esempio utilizzando [UpdateFleetAttributes](#) o [UpdateContainerFleet](#)) non modifica la versione dell'AMI. È necessario creare una nuova flotta.

- Sostituisci regolarmente le flotte in base alla sicurezza e allo stato di salute: imposta un programma regolare per creare nuove flotte e ritirare le vecchie flotte. Prendi in considerazione l'utilizzo di un servizio come [Amazon Q](#) per rivedere il codice di gioco con l'attuale versione AMI, rilevare problemi di sicurezza e suggerire passaggi correttivi.
- Verifica le build del server con le versioni AMI più recenti prima della distribuzione: potrebbe essere necessario modificare la build del server e caricarla Amazon GameLift Servers prima di creare una nuova flotta.
- Gestisci le quote della flotta per il tuo AWS account: puoi richiedere aumenti dei limiti, se necessario, per creare flotte sostitutive. Per ulteriori informazioni, consulta [Endpoint e quote di Amazon GameLift Servers](#).
- Valuta la possibilità di automatizzare la sostituzione della flotta: puoi automatizzare i processi per creare nuove flotte e migrare il traffico dei giocatori dalle flotte più vecchie. Esempio:
 - Utilizzalo per automatizzare AWS CloudFormation la creazione e la gestione della flotta. Mantieni le configurazioni della flotta come CloudFormation modelli e usale per avviare stack di risorse.
 - Sfrutta la funzione Amazon GameLift Servers alias per astrarre una flotta specifica. IDs Gli alias della flotta consentono di spostare facilmente il traffico dei giocatori da una flotta esistente a una nuova senza interrompere le sessioni di gioco in corso. Per informazioni dettagliate, vedi [Designazione astratta e Amazon GameLift Servers della flotta con un alias](#).
 - Utilizza strategie blue/green di implementazione per ridurre il rischio di migrazione e mantenere zero i tempi di inattività. Con due ambienti di produzione identici, puoi trarre vantaggio da un

ambiente di test completo simile alla produzione, esercitare un maggiore controllo sul processo di migrazione e garantire ripristini immediati.

Proteggi le configurazioni delle porte

Consigliamo vivamente di non aprire porte a Internet perché ciò comporta un rischio per la sicurezza. Ad esempio, la seguente configurazione apre una porta desktop remota che consente a chiunque su Internet di accedere all'istanza:

```
{
  "FleetId": "<fleet identifier>",
  "InboundPermissionAuthorizations": [
    {
      "FromPort": 3389,
      "IpRange": "0.0.0.0/0",
      "Protocol": "RDP",
      "ToPort": 3389
    }
  ]
}
```

Utilizza invece [UpdateFleetPortSettings](#) per aprire una porta con un indirizzo IP o un intervallo di indirizzi specifico, come mostrato in questo esempio:

```
{
  "FleetId": "<fleet identifier>",
  "InboundPermissionAuthorizations": [
    {
      "FromPort": 3389,
      "IpRange": "54.186.139.221/32",
      "Protocol": "TCP",
      "ToPort": 3389
    }
  ]
}
```

Risorse di sicurezza aggiuntive

Per ulteriori informazioni su come rendere Amazon GameLift Servers più sicuro il proprio utilizzo, consulta il [pilastro AWS Well-Architected Tool Sicurezza](#).

Amazon GameLift Servers pianificazione dei prezzi e dei costi

Comprendere la struttura Amazon GameLift Servers dei prezzi e implementare strategie convenienti nella soluzione di hosting di giochi è essenziale per ottimizzare il budget dell'infrastruttura di gioco.

[Per informazioni dettagliate sull'assistenza, inclusi esempi di scenari di prezzo e strumenti di determinazione dei prezzi, consulta la sezione Prezzi. Amazon GameLift Servers](#)

AWS I nuovi clienti possono utilizzarlo Amazon GameLift Servers senza incorrere in addebiti nell'ambito del piano gratuito per un massimo di 12 mesi. Resta entro i limiti di utilizzo del piano gratuito per evitare addebiti.

Modelli tariffari

Amazon GameLift Servers offre diversi modelli di prezzo progettati per soddisfare diversi scenari di utilizzo ed esigenze aziendali. Ogni modello ha una base di costo diversa. È possibile controllare i costi regolando i fattori di costo in ciascun modello.

Hosting gestito

Le flotte di container gestite EC2 e gestite utilizzano EC2 istanze Amazon che Amazon GameLift Servers gestiscono. Quando configuri l'hosting di giochi, scegli il tipo di EC2 istanza Amazon da utilizzare.

- Costi di calcolo: paga in base all'utilizzo orario dell' EC2 istanza per ospitare sessioni di gioco. I fattori di costo per le istanze includono:
 - AWS Regione
 - Famiglia, tipo e dimensione dell'istanza
 - Uso delle istanze Spot rispetto alle istanze On-Demand
 - Sistema operativo (le flotte di container gestite funzionano solo su Linux)
- Trasferimento dati: paga per il traffico tra client di gioco e server di gioco ospitati

Hosting ovunque

L'hosting Anywhere viene eseguito su risorse di hosting di giochi fornite e gestite da te, quindi non ci sono costi di elaborazione. I costi si basano invece su:

- Sessioni di gioco: paga in base al numero di sessioni di gioco effettuate sui computer Anywhere
- Minuti di connessione: vengono pagati i minuti di connessione al processo server

Utilizza il calcolatore dei prezzi per Amazon GameLift Servers stimare i costi per diverse Amazon GameLift Servers configurazioni. Usa il calcolatore per progettare la tua soluzione di hosting di giochi o per prevedere i costi delle modifiche alla configurazione. Vedi [Generare stime Amazon GameLift Servers dei prezzi](#).

Genera stime Amazon GameLift Servers dei prezzi

Con Calcolatore dei prezzi AWS, puoi [creare una stima dei prezzi per Amazon GameLift Servers](#). Non è necessaria una conoscenza approfondita Account AWS o approfondita AWS per utilizzare la calcolatrice.

Calcolatore dei prezzi AWS calculator ti guida attraverso le decisioni che influiscono sui costi del servizio per darti un'idea di quanto Amazon GameLift Servers potrebbe costare il tuo progetto di gioco. Se non sei ancora sicuro di come intendi utilizzarlo Amazon GameLift Servers, utilizza i valori predefiniti per generare una stima. Quando pianifichi l'utilizzo in produzione, la calcolatrice può aiutarti a testare potenziali scenari e generare stime più accurate.

Stima dell'Amazon GameLift Servers hosting gestito

Questa opzione fornisce una stima dei costi per l'hosting dei giochi su server Amazon GameLift Servers gestiti, compresi i costi per l'utilizzo delle istanze del server e il trasferimento dei dati. Con l'hosting Amazon GameLift Servers gestito, non ci sono costi aggiuntivi per il FlexMatch matchmaking.

Se ospiti o prevedi di ospitare server di gioco in più di una AWS regione o su più di un tipo di istanza, crea una stima per ogni regione e tipo di istanza.

Istanze Amazon GameLift Servers

Questa sezione ti aiuta a stimare il tipo e il numero di risorse di calcolo necessarie per ospitare sessioni di gioco per i tuoi giocatori. Amazon GameLift Servers utilizza [istanze Amazon Elastic Compute Cloud \(Amazon EC2\)](#) per gestire i server di gioco. Nel Amazon GameLift Servers, distribuisce una flotta di istanze con un tipo di istanza e un sistema operativo specifici. Se disponi o prevedi di avere più flotte, crea una stima per ciascuna flotta.

Per iniziare, apri la [Amazon GameLift Servers pagina Configura](#) di Calcolatore dei prezzi AWS. Aggiungi una descrizione, scegli una regione, quindi scegli Estimate Amazon GameLift Servers hosting (Instance + Data Transfer Out). In Amazon GameLift ServersIstanze, completa i seguenti campi:

- Numero massimo di giocatori concorrenti (picco di potenza della CCU)

Questo è il numero massimo di giocatori che possono connettersi contemporaneamente ai tuoi server di gioco. Questo campo indica la quantità di capacità di hosting Amazon GameLift Servers necessaria per soddisfare la domanda di picco dei giocatori. Inserisci il numero massimo giornaliero di giocatori che prevedi di ospitare utilizzando le istanze nella AWS regione prescelta.

Ad esempio, se desideri consentire a 1.000 giocatori di connettersi al tuo gioco contemporaneamente, mantieni il valore predefinito di **1000**.

- CCU media all'ora come percentuale della CCU giornaliera di picco

Questo è il numero medio di giocatori simultanei all'ora in un periodo di 24 ore. Utilizziamo questo valore per stimare la quantità di capacità di hosting sostenuta da Amazon GameLift Servers mantenere per i tuoi giocatori. Se non sei sicuro del valore percentuale da utilizzare, mantieni il valore predefinito della **50** percentuale. Per i giochi con una domanda stabile da parte dei giocatori, ti consigliamo di inserire un valore **70** percentuale.

Ad esempio, se il gioco ha una CCU oraria media di 6.000 e una CCU di picco di 10.000, inserisci il valore percentuale. **60**

- Sessioni di gioco per esempio

Questo è il numero di sessioni di gioco che ciascuna istanza del server di gioco può ospitare contemporaneamente. I fattori che possono influire su questo numero includono il fabbisogno di risorse del server di gioco, il numero di giocatori da ospitare in ogni sessione di gioco e le aspettative in termini di prestazioni dei giocatori. Se conosci il numero di sessioni di gioco simultanee per il tuo gioco, inserisci quel valore. In alternativa, mantieni il valore predefinito di **20**

- Giocatori per sessione di gioco

Questo è il numero medio di giocatori che si connettono a una sessione di gioco, come definito nella progettazione del gioco. Se utilizzi modalità di gioco con un numero diverso di giocatori, calcola un numero medio di giocatori per sessione di gioco nell'arco dell'intera partita. Il valore predefinito è **8**.

- Buffer di inattività dell'istanza%

Questa è la percentuale di capacità di hosting inutilizzata da mantenere come riserva per gestire i picchi improvvisi della domanda da parte dei giocatori. La dimensione del buffer è una percentuale del numero totale di istanze in un parco istanze. Il valore predefinito è **10** percentuale.

Ad esempio, con un buffer di inattività del 20%, un parco istanze che supporta giocatori con 100 istanze attive mantiene 20 istanze inattive.

- **Istanza Spot%**

Amazon GameLift Servers le flotte possono utilizzare una combinazione di istanze On-Demand e istanze Spot. Mentre le istanze on demand offrono una disponibilità più affidabile, le istanze Spot offrono un'alternativa altamente efficiente in termini di costi. Ti consigliamo di utilizzare una combinazione per ottimizzare sia il risparmio sui costi che la disponibilità. Per informazioni su come Amazon GameLift Servers utilizza le istanze Spot, consulta [On-Demand Instances versus Spot Instances](#).

Per questo campo, inserisci la percentuale di istanze Spot da gestire in un parco istanze. Consigliamo una percentuale di istanze Spot compresa tra il 50 e l'85 per cento. Il valore predefinito è **50** percentuale.

Ad esempio, se si implementa un parco istanze con 100 istanze e si specifica la **40** percentuale, si Amazon GameLift Servers adopera per mantenere 60 istanze on demand e 40 istanze Spot.

- **Tipo di istanza**

Amazon GameLift Servers le flotte possono utilizzare una gamma di tipi di EC2 istanze Amazon che variano in termini di potenza di calcolo, memoria, archiviazione e capacità di rete. Quando configuri una Amazon GameLift Servers flotta, scegli il tipo di istanza più adatto alle esigenze del tuo gioco. Per informazioni sulla selezione di un tipo di istanza con Amazon GameLift Servers, consulta [Scegli le risorse di elaborazione per una flotta gestita](#).

Se conosci il tipo di istanza che stai utilizzando o intendi utilizzare nel tuo Amazon GameLift Servers parco istanze, scegli quel tipo. Se non sei sicuro del tipo da scegliere, valuta la possibilità di scegliere c5.large. Si tratta di un tipo ad alta disponibilità con dimensioni e funzionalità medie.

- **Sistema operativo**

Questo campo specifica il sistema operativo su cui girano i server di gioco, Linux o Windows. Il valore predefinito è Linux.

Trasferimento dati in uscita (DTO)

Questa sezione ti aiuta a stimare il costo del traffico tra i tuoi client di gioco e i server di gioco. Le tariffe per il trasferimento dei dati si applicano solo al traffico in uscita. Il trasferimento dei dati in entrata non ha costi.

Nella [Amazon GameLift Servers pagina Configura](#) di Calcolatore dei prezzi AWS, espandi Data transfer out (DTO), quindi completa i seguenti campi:

- Tipo di stima DTO

Puoi scegliere di stimare il DTO in uno dei due modi seguenti, a seconda di come tieni traccia del trasferimento dei dati durante il gioco.

- Al mese (in GB): se registri il traffico mensile per i tuoi server di gioco, scegli questo tipo.
- Per giocatore: se tieni traccia del trasferimento di dati per giocatore, scegli questo tipo. Questo è il tipo predefinito.

Nel campo seguente, si stima il DTO per giocatore in base al numero di ore di gioco calcolato nella sezione precedente.

- DTO al mese (in GB)

Se hai scelto il tipo di stima DTO mensile (in GB), inserisci l'utilizzo mensile stimato del DTO in GB per ogni istanza, per regione.

- DTO per giocatore

Se hai scelto il tipo di stima del DTO per giocatore, inserisci l'utilizzo stimato del DTO per giocatore per giocatore in KB/sec. Il valore predefinito è **4**.

Quando hai finito di configurare la stima del Amazon GameLift Servers prezzo, scegli Aggiungi alla mia stima. Per ulteriori informazioni sulla creazione e la gestione delle stime in Calcolatore dei prezzi AWS, consulta [Creare una stima, configurare un servizio e aggiungere altri servizi](#) nella Guida per l'Calcolatore dei prezzi AWS utente.

Strategie di ottimizzazione dei costi

Utilizza queste strategie con una soluzione di hosting gestito per ridurre i costi del cloud hosting mantenendo al contempo prestazioni e esperienza di gioco elevate.

Best practice

Segui questi suggerimenti per incorporare pratiche di ottimizzazione dei costi per l'hosting dei tuoi giochi. Come minimo, ti consigliamo di pianificare revisioni periodiche dei Amazon GameLift Servers costi e dei modelli di utilizzo. Se hai domande relative ai prezzi, contatta il

Configurazione del parco istanze

- Usa il calcolatore dei prezzi: utilizza questo strumento con i dati di hosting dei giochi per esplorare le opzioni e testare potenziali scenari di configurazione per risparmiare sui costi. Vedi il [calcolatore AWS dei prezzi](#) per. Amazon GameLift Servers
- Risparmia sullo sviluppo: per lo sviluppo e il test, utilizza tipi di istanze più piccoli e meno costosi ed esegui solo quando necessario.
- Abbina il tipo di istanza al gioco: scegli le famiglie, i tipi e le dimensioni di EC2 istanze Amazon per le tue flotte più adatte ai requisiti del tuo gioco. Consulta la sezione Tipi di istanze in [Scegli le risorse di calcolo per una flotta gestita](#).
- Usa i tipi di istanze Graviton: le istanze Graviton sono alimentate da processori basati su ARM. Offrono un miglior rapporto prezzo/prestazioni, una maggiore efficienza energetica e costi inferiori rispetto alle istanze analoghe basate su x86 con il modello di prezzo On-Demand.
- Usa le istanze Spot: le istanze Spot hanno generalmente un costo inferiore rispetto a quelle on demand, ma potrebbero non essere sempre disponibili. Combina flotte Spot e On-Demand per bilanciare basso costo e alta disponibilità (si consiglia l'utilizzo di Spot al 50-85%). Usa le istanze Spot per modalità di gioco non critiche o durante le ore non di punta. Per monitorare la fattibilità delle istanze Spot, tieni traccia di metriche come `e. InstanceInterruptions` `GameServerInterruptions` Per informazioni, consulta [On-Demand Instances versus Spot Instances](#).
- Esegui server su Linux: crea il runtime del tuo server di gioco per Linux. Le istanze distribuite con Linux sono generalmente più convenienti rispetto a quelle con Windows.
- Ottimizza l'utilizzo delle risorse: configura le flotte per la massima efficienza eseguendo il maggior numero possibile di processi simultanei sui server di gioco mantenendo al contempo le prestazioni. Per informazioni, consulta [Ottimizza la configurazione del runtime del server di gioco su server gestiti Amazon GameLift Servers](#).
- Crea una strategia multiregionale: l'implementazione dell'hosting di giochi in più sedi può offrire ai giocatori una latenza inferiore e migliori opzioni di backup. Equilibra la copertura regionale e l'efficienza in termini di costi implementando innanzitutto l'hosting nei principali mercati di

riferimento e utilizzando sedi secondarie per aumentare la capacità nelle ore di punta. Per informazioni, consulta [Crea una coda con più sedi](#).

Scalabilità della capacità della flotta

- Scalabilità fino a zero: quando le flotte non sono in uso, imposta manualmente la capacità della flotta su zero per evitare addebiti inutili. Per informazioni, consulta [Imposta manualmente la capacità per una Amazon GameLift Servers flotta](#).
- Aggiungi la scalabilità automatica: evita il sovradimensionamento delle risorse di hosting utilizzando la scalabilità automatica per regolare la capacità di hosting dei giochi. Adatta la capacità alla fluttuazione della domanda dei giocatori e ad altre metriche chiave. Per informazioni, consulta [Capacità di dimensionamento automatico del parco istanze con Amazon GameLift Servers](#).
- Mantieni un buffer: per gestire i picchi improvvisi della domanda dei giocatori senza farli attendere, utilizza il tracciamento degli obiettivi per mantenere un buffer di server di gioco inattivi. Personalizza il buffer in base alle dimensioni e ai modelli di utilizzo della tua base di giocatori. Per informazioni, consulta [Dimensionamento automatico basato sugli obiettivi](#).

Posizionamento della sessione di gioco

- Utilizza le code di posizionamento: le Amazon GameLift Servers code si basano su un algoritmo per determinare le «migliori» sedi di hosting possibili per le sessioni di gioco, in base al costo delle risorse di hosting e ad altri fattori. Per informazioni, consulta [Configura il posizionamento delle sessioni di gioco](#).
- Personalizza le priorità delle code: puoi modificare il modo in cui una coda assegna la priorità ai costi di hosting quando organizza le sessioni di gioco. Per informazioni, consulta [Dare la priorità al posizionamento delle sessioni di gioco](#).
- Monitora le metriche di posizionamento: monitora il modo in cui le code scelgono le posizioni per le sessioni di gioco per identificare le opportunità di ottimizzazione. Per informazioni, consulta [Parametri Amazon GameLift Servers per code](#).

Trasferimento dei dati

- Tieni traccia dei trasferimenti di dati: monitora il modo in cui i dati vengono trasferiti tra i client di gioco e i server e adotta misure per ottimizzare l'attività.

- Usa la compressione dei dati: prendi in considerazione l'implementazione di tecniche di compressione dei dati per i trasferimenti di dati tra client di gioco e server. La compressione svolge un ruolo cruciale nella riduzione dell'utilizzo della larghezza di banda e nel miglioramento delle prestazioni e della latenza di gioco.

Costi delle risorse e strumenti di utilizzo

Scopri come utilizzare AWS gli strumenti per monitorare e ottimizzare i costi di hosting dei giochi con Amazon GameLift Servers. Per informazioni sugli strumenti aggiuntivi, consulta [Gestione dei costi e fatturazione AWS](#).

AWS strumenti di gestione dei costi

- Console di fatturazione: verifica le AWS fatture e l'utilizzo. Vedi [Configurazione](#) della fatturazione.
- Avvisi sull'utilizzo del piano gratuito: imposta le notifiche quando si raggiungono i limiti del piano gratuito. Vedi [Monitoraggio dell'utilizzo del piano AWS gratuito](#).
- Avvisi di CloudWatch fatturazione Amazon: configura avvisi quando l'utilizzo raggiunge soglie personalizzate. Consulta [Creare un allarme di fatturazione](#) per monitorare gli addebiti stimati. AWS

Allocazione e monitoraggio dei costi

- Tag di allocazione dei costi: contrassegna le tue flotte e altre risorse per organizzare e tenere traccia dei costi di hosting. Vedi [Organizzazione e monitoraggio dei costi utilizzando i tag di allocazione AWS dei costi](#).
- Rapporti sui costi: crea report classificati in base ai tag assegnati. Vedere [Utilizzo del rapporto mensile sull'allocazione dei costi](#).
- AWS Cost Explorer— Analizza costi, tendenze e previsioni con filtri e visualizzazioni personalizzabili. Vedi [Analisi dei costi e dell'utilizzo](#) con. AWS Cost Explorer
- Budget AWS— Monitora e agisci sui AWS costi e sull'utilizzo. Vedi [Gestione dei costi con Budget AWS](#).

Monitoraggio delle prestazioni con Amazon GameLift Servers e Amazon CloudWatch

Monitora queste metriche chiave per ottimizzare l'utilizzo delle risorse. Visualizza i parametri nella Amazon GameLift Servers console o utilizza i CloudWatch dashboard di Amazon. Per i dettagli su

tutte le metriche disponibili per Amazon GameLift Servers, consulta. [Monitora Amazon GameLift Servers con Amazon CloudWatch](#)

- Metriche delle istanze: `ActiveInstances` `IdleInstances` `PercentIdleInstances`
- Metriche dei processi del server: `ActiveServerProcesses` `HealthyServerProcesses`
- Metriche delle sessioni di gioco: `ActiveGameSessions` `AvailableGameSessions`
- Metriche della sessione dei giocatori: `CurrentPlayerSessions`
- Metriche della coda: `AverageWaitTime` `QueueDepth`
- Metriche di matchmaking — `CurrentTickets` `MatchesPlaced`
- Prestazioni hardware
— `CPUUtilization`, `NetworkIn`, `NetworkOut`, `DiskReadBytes`, `DiskWriteBytes`
- Parametri di utilizzo delle risorse
 - Utilizzato `PercentIdleInstances` per determinare la dimensione ottimale del parco veicoli.
 - Tieni traccia `PercentAvailableGameSessions` per garantire una capacità di hosting dei giochi sufficiente.
 - Monitora `InstanceInterruptions` e determina `GameServerInterruptions` la fattibilità delle istanze Spot.

Amazon GameLift Serversguide di riferimento

Questa sezione contiene la documentazione di riferimento per l'utilizzo Amazon GameLift Servers.

Argomenti

- [API di servizio per Amazon GameLift Servers](#)
- [Server SDK 5.x per Amazon GameLift Servers](#)
- [Server SDK per la Amazon GameLift Servers versione 4 e precedenti](#)
- [Eventi di collocamento delle sessioni di gioco](#)
- [Amazon GameLift ServersVersioni AMI](#)
- [Amazon GameLift Serverssedi di assistenza](#)
- [Endpoint e quote di Amazon GameLift Servers](#)
- [Limiti delle API di Amazon GameLift Servers](#)
- [Limiti delle API SDK di Amazon GameLift Servers](#)
- [Amazon GameLift ServersFari ping UDP](#)

API di servizio per Amazon GameLift Servers

Usa questo elenco basato sulle attività per trovare le operazioni API durante la creazione di soluzioni di hosting di Amazon GameLift Servers giochi e altre funzionalità. L' AWS SDK include queste operazioni nel namespace. `aws.gamelift` [Scarica l' AWS SDK](#) o [visualizza la documentazione di riferimento dell'Amazon GameLift ServersAPI](#). Puoi anche utilizzare l'API con l'interfaccia a riga di AWS comando (AWS CLI), come documentato nel riferimento ai [AWS CLI comandi](#).

L'API include due serie di operazioni per l'hosting gestito di giochi:

- [Gestisci le risorse Amazon GameLift Servers di hosting](#)
- [Avvia sessioni di gioco e unisciti ai giocatori](#)

L'API Amazon GameLift Servers di servizio contiene anche operazioni da utilizzare con altri Amazon GameLift Servers strumenti e soluzioni. Per un elenco di FleetIQ APIs, consulta [Operazioni FleetIQ API](#). Per un elenco di FlexMatch APIs matchmaking, vedi [Operazioni FlexMatch API](#).

Gestisci le risorse Amazon GameLift Servers di hosting

Richiama queste operazioni per configurare le risorse di hosting per i tuoi server di gioco, scalare la capacità per soddisfare la domanda dei giocatori, ottenere metriche di prestazioni e utilizzo e altro ancora. Utilizza queste operazioni API per ospitare server di gioco con Amazon GameLift Servers, incluso Amazon GameLift Servers Realtime. Puoi anche lavorare [Amazon GameLift Servers sulla console](#) per la maggior parte delle attività di gestione delle risorse oppure puoi effettuare chiamate con lo strumento AWS Command Line Interface (AWS CLI).

Prepara i server di gioco per la distribuzione

Carica e configura il codice del server di gioco del gioco in preparazione della distribuzione e del lancio sulle risorse di hosting.

Gestisci build di server di gioco personalizzate

- [upload-build](#): carica i file di build da un percorso locale e crea una nuova Amazon GameLift Servers risorsa di build. Questa operazione, disponibile come AWS CLI comando, è il modo più comune per caricare build di server di gioco.
- [CreateBuild](#)— Crea una nuova build utilizzando file archiviati in un bucket Amazon S3.
- [ListBuilds](#)— Ottieni un elenco di tutte le build caricate in una regione. Amazon GameLift Servers
- [DescribeBuild](#)— Recupera le informazioni associate a una build.
- [UpdateBuild](#)— Modifica i metadati della build, inclusi il nome e la versione della build.
- [DeleteBuild](#)— Rimuovi una build da Amazon GameLift Servers.

Gestisci gli script Amazon GameLift Servers Realtime di configurazione

- [CreateScript](#)— Carica JavaScript file e crea una nuova risorsa di Amazon GameLift Servers script.
- [ListScripts](#)— Ottieni un elenco di tutti gli Realtime script caricati in una Amazon GameLift Servers regione.
- [DescribeScript](#)— Recupera le informazioni associate a uno Realtime script.
- [UpdateScript](#)— Modifica i metadati dello script e carica il contenuto dello script rivisto.
- [DeleteScript](#)— Rimuovere uno Realtime script da. Amazon GameLift Servers

Configura le risorse informatiche per l'hosting

Configura le risorse di hosting e distribuiscile con lo script di build o Realtime configurazione del server di gioco.

Crea e gestisci flotte

- [CreateFleet](#)— Configura e distribuisci una nuova Amazon GameLift Servers flotta di risorse informatiche per far funzionare i tuoi server di gioco. Una volta distribuiti, i server di gioco vengono avviati automaticamente così come configurati e pronti per ospitare sessioni di gioco.
- [ListFleets](#)— Ottieni un elenco di tutte le flotte in una Amazon GameLift Servers regione.
- [DeleteFleet](#)— Rimuovi una flotta che non utilizza più server di gioco o ospita giocatori.
- Visualizza/aggiorna le posizioni della flotta.
 - [CreateFleetLocations](#)— Aggiungi sedi remote a una flotta esistente che supporta più sedi
 - [DescribeFleetLocationAttributes](#)— Ottieni un elenco di tutte le postazioni remote di una flotta e visualizza lo stato attuale di ciascuna località.
 - [DeleteFleetLocations](#)— Rimuovi le postazioni remote da una flotta che supporta più sedi.
- Visualizzare/aggiornare le configurazioni del parco istanze.
 - [DescribeFleetAttributes/UpdateFleetAttributes](#)— Visualizza o modifica i metadati e le impostazioni di una flotta per la protezione delle sessioni di gioco e i limiti di creazione di risorse.
 - [DescribeFleetPortSettings/UpdateFleetPortSettings](#)— Visualizza o modifica le autorizzazioni in entrata (indirizzi IP e intervalli di impostazione delle porte) consentite per un parco veicoli.
 - [DescribeRuntimeConfiguration/UpdateRuntimeConfiguration](#)— Visualizza o modifica quali processi server (e quanti) eseguire su ciascuna istanza di un parco istanze.

Gestisci la capacità del parco veicoli

- [Descrivi EC2 InstanceLimits](#): recupera il numero massimo di istanze consentite per l' AWS account corrente e il livello di utilizzo corrente.
- [DescribeFleetCapacity](#)— Recupera le impostazioni di capacità correnti per la regione di origine di una flotta.
- [DescribeFleetLocationCapacity](#)— Recupera le impostazioni di capacità correnti per ogni sede di una flotta con più sedi.
- [UpdateFleetCapacity](#)— Regola manualmente le impostazioni di capacità per una flotta.

- Configurazione:

- [PutScalingPolicy](#)— Attiva la scalabilità automatica basata sulla destinazione o crea una politica di scalabilità automatica personalizzata o aggiorna una politica esistente.
- [DescribeScalingPolicies](#)— Recuperare una politica di autoscaling esistente.
- [DeleteScalingPolicy](#)— Eliminare una politica di scalabilità automatica e impedire che influisca sulla capacità del parco veicoli.
- [StartFleetActions](#)— Riavvia le politiche di scalabilità automatica di una flotta.
- [StopFleetActions](#)— Sospendere le politiche di scalabilità automatica di una flotta.

Monitorare l'attività del parco istanze.

- [DescribeFleetUtilization](#)— Recupera le statistiche sul numero di processi del server, sessioni di gioco e giocatori attualmente attivi su una flotta.
- [DescribeFleetLocationUtilization](#)— Recupera le statistiche di utilizzo per ogni località in una flotta con più sedi.
- [DescribeFleetEvents](#)— Visualizza gli eventi registrati per una flotta durante un periodo di tempo specificato.
- [DescribeGameSessions](#)— Recupera i metadati della sessione di gioco, tra cui la durata di una partita e il numero attuale di giocatori.

Imposta le code per il posizionamento delle sessioni di gioco

Configura le code in più regioni e in più parchi istanze per posizionare le sessioni di gioco con le migliori risorse di hosting disponibili per costo, latenza e resilienza.

- [CreateGameSessionQueue](#)— Crea una coda da utilizzare durante l'elaborazione delle richieste di posizionamento nelle sessioni di gioco.
- [DescribeGameSessionQueues](#)— Recupera le code delle sessioni di gioco definite in una regione.
Amazon GameLift Servers
- [UpdateGameSessionQueue](#)— Modifica la configurazione di una coda di sessione di gioco.
- [DeleteGameSessionQueue](#)— Rimuove una coda di sessione di gioco dalla regione.

Gestione di alias

Utilizza gli alias per rappresentare i parchi istanze o per creare una destinazione alternativa del terminale. Gli alias sono utili per la transizione dell'attività di gioco da un parco istanze all'altro, ad esempio durante gli aggiornamenti della build del server di gioco.

- [CreateAlias](#)— Definisci un nuovo alias e, facoltativamente, assegnalo a una flotta.
- [ListAliases](#)— Ottieni tutti gli alias della flotta definiti in una regione. Amazon GameLift Servers
- [DescribeAlias](#)— Recupera informazioni su un alias esistente.
- [UpdateAlias](#)— Modifica le impostazioni di un alias, ad esempio reindirizzandolo da una flotta all'altra.
- [DeleteAlias](#)— Rimuovere un alias dalla regione.
- [ResolveAlias](#)— Ottieni l'ID della flotta a cui punta un alias specificato.

Connect a istanze di hosting gestite

Visualizza le informazioni sulle singole istanze di un parco istanze o richiedi l'accesso remoto a una specifica istanza del parco istanze per la risoluzione dei problemi.

- [DescribeInstances](#)— Ottieni informazioni su ogni istanza del parco istanze, tra cui ID dell'istanza, indirizzo IP, posizione e stato.
- [GetInstanceAccess](#)— Richiedere le credenziali di accesso necessarie per connettersi in remoto a un'istanza specifica in un parco istanze.

Configurazione del peering VPC

Crea e gestisci connessioni peering VPC tra le tue risorse di Amazon GameLift Servers hosting e altre risorse. AWS

- [CreateVpcPeeringAuthorization](#)— Autorizza una connessione peering a uno dei tuoi VPCs
- [DescribeVpcPeeringAuthorizations](#)— Recupera autorizzazioni di connessione peering valide.
- [DeleteVpcPeeringAuthorization](#)— Eliminare l'autorizzazione di una connessione peering.
- [CreateVpcPeeringConnection](#)— Stabilisci una connessione peering tra il VPC di Amazon GameLift Servers una flotta e uno dei tuoi VPCs
- [DescribeVpcPeeringConnections](#)— Recupera informazioni sulle connessioni peering VPC attive o in sospeso con una flotta. Amazon GameLift Servers

- [DeleteVpcPeeringConnection](#)— Eliminare una connessione peering VPC con una flotta. Amazon GameLift Servers

Avvia sessioni di gioco e unisciti ai giocatori

Richiama queste operazioni da un servizio di backend per avviare nuove sessioni di gioco, ottenere informazioni sulle sessioni di gioco esistenti e unire i giocatori alle sessioni di gioco. Queste operazioni sono destinate all'uso con server di gioco personalizzati ospitati su Amazon GameLift Servers. Se lo utilizzi Amazon GameLift ServersRealtime, gestisci le sessioni di gioco utilizzando [l'API Amazon GameLift ServersRealtime client \(C#\)](#).

- Avviare nuove sessioni di gioco per uno o più giocatori.
 - [StartGameSessionPlacement](#)— Chiedi Amazon GameLift Servers di trovare le migliori risorse di hosting disponibili e di iniziare una nuova sessione di gioco. Questo è il metodo preferito per creare nuove sessioni di gioco. Si basa sulle code delle sessioni di gioco per tenere traccia della disponibilità di hosting in più regioni e utilizza FleetIQ algoritmi per dare priorità ai posizionamenti in base alla latenza dei giocatori, ai costi di hosting, alla posizione, ecc.
 - [DescribeGameSessionPlacement](#)— Ottieni i dettagli e lo stato di una richiesta di collocamento.
 - [StopGameSessionPlacement](#)— Annullare una richiesta di collocamento.
 - [CreateGameSession](#)— Inizia una nuova sessione di gioco vuota in una posizione specifica della flotta. Questa operazione ti dà un maggiore controllo su dove iniziare la sessione di gioco, anziché utilizzarla FleetIQ per valutare le opzioni di posizionamento. È necessario aggiungere giocatori alla nuova sessione di gioco in un passaggio separato.
- Attira i giocatori nelle sessioni di gioco esistenti. Trova sessioni di gioco in corso con gli slot disponibili e riservale ai nuovi giocatori.
 - [CreatePlayerSession](#)— Riserva uno slot libero per consentire a un giocatore di partecipare a una sessione di gioco.
 - [CreatePlayerSessions](#)— Riserva slot aperti per consentire a più giocatori di partecipare a una sessione di gioco.
- Utilizzare dati sulle sessioni di gioco e dei giocatori. Gestisci le informazioni sulle sessioni di gioco e sulle sessioni dei giocatori.
 - [SearchGameSessions](#)— Richiedi un elenco di sessioni di gioco attive in base a una serie di criteri di ricerca.
 - [DescribeGameSessions](#)— Recupera i metadati per sessioni di gioco specifiche, tra cui la durata di attività e il numero attuale di giocatori.

- [DescribeGameSessionDetails](#)— Recupera i metadati, inclusa l'impostazione di protezione della sessione di gioco, per una o più sessioni di gioco.
- [DescribePlayerSessions](#)— Ottieni dettagli sull'attività dei giocatori, tra cui stato, tempo di gioco e dati del giocatore.
- [UpdateGameSession](#)— Modifica le impostazioni della sessione di gioco, ad esempio il numero massimo di giocatori e la politica di iscrizione.
- [GetGameSessionLogUrl](#)— Ottieni la posizione dei log salvati per una sessione di gioco.

Server SDK 5.x per Amazon GameLift Servers

Questa sezione fornisce la documentazione di riferimento per il server SDK 5.x for. Amazon GameLift Servers L'SDK del server fornisce le funzionalità di base utilizzate dal server di gioco per interagire con il servizio. Amazon GameLift Servers Ad esempio, il server di gioco riceve dal servizio richieste di avvio e interruzione delle sessioni di gioco e fornisce aggiornamenti regolari sullo stato delle sessioni di gioco al servizio. Integra i tuoi server di gioco con l'SDK del server prima di distribuirli per l'hosting.

Usa questo riferimento all'SDK del server per integrare i tuoi server di gioco multiplayer personalizzati con cui ospitare. Amazon GameLift Servers Per indicazioni sul processo di integrazione, consulta [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#).

L'ultima versione principale del server SDK for Amazon GameLift Servers è la 5.x. Le seguenti funzionalità di hosting richiedono l'uso della versione 5.x:

- Amazon GameLift ServersOvunque
- Amazon GameLift Serversplugin per Unreal Engine e Unity

Note

Se devi utilizzare la versione 4.x o precedente dell'SDK del server, consulta la documentazione e le informazioni [Server SDK per la Amazon GameLift Servers versione 4 e precedenti](#) per il download.

Argomenti

- [Aggiornamenti nel server SDK 5 per Amazon GameLift Servers](#)
- [Esegui la migrazione al server SDK 5.x per Amazon GameLift Servers](#)

- [Server C++ SDK 5.x per -- Azioni Amazon GameLift Servers](#)
- [Server C# SDK 5.x per -- Azioni Amazon GameLift Servers](#)
- [Go server SDK per Amazon GameLift Servers -- Azioni](#)
- [Server C++ \(Unreal\) SDK 5.x per -- Azioni Amazon GameLift Servers](#)

Aggiornamenti nel server SDK 5 per Amazon GameLift Servers

I server di gioco ospitati utilizzano l'SDK del server Amazon GameLift Servers per comunicare con il Amazon GameLift Servers servizio e avviare e gestire le sessioni di gioco per i giocatori. L'ultima versione, Amazon GameLift Servers server SDK 5, offre una serie di miglioramenti e supporto per nuove Amazon GameLift Servers funzionalità. Se la versione del tuo server di gioco utilizza attualmente Amazon GameLift Servers server SDK 4 o versioni precedenti, segui le indicazioni riportate in questo argomento per aggiornare i tuoi giochi.

Amazon GameLift Servers la versione 5.0.0 e successive dell'SDK del server include questi aggiornamenti:

- Linguaggi espansi: le librerie sono disponibili nelle seguenti lingue: C++, C#, Go. Puoi creare le librerie C++ da utilizzare con Unreal Engine.
- Supporto per i plug-in del motore di gioco: i plug-in Amazon GameLift Servers autonomi per Unreal Engine e Unity richiedono Amazon GameLift Servers le librerie SDK 5 del server. Questi plugin offrono flussi di lavoro guidati per l'integrazione, il test e la distribuzione dei giochi per l'hosting. Amazon GameLift Servers Vedi e documentazione. [Amazon GameLift Servers plugin per Unity \(server SDK 5.x\)](#) [Amazon GameLift Servers plugin per Unreal Engine](#)
- Amazon GameLift Servers Supporto ovunque: con le flotte Anywhere puoi configurare le tue risorse di hosting per utilizzare le Amazon GameLift Servers funzionalità (incluso il matchmaking). Aggiungi l'Amazon GameLift Servers agente per automatizzare la gestione del ciclo di vita delle sessioni di gioco. Usa le flotte Anywhere per l'hosting di produzione con hardware locale o configura ambienti di test per un rapido sviluppo iterativo di giochi. [See Hosting ovunque and the Agent. Amazon GameLift Servers](#)
- Strumenti di test aggiornati: la funzione Amazon GameLift Servers Anywhere ti consente di configurare ambienti di test locali o basati sul cloud per i tuoi giochi. Configura i test con o senza l'Amazon GameLift Servers agente. Questi strumenti sostituiscono Amazon GameLift Servers Local. Consultare [Configura i test locali con Amazon GameLift Servers Anywhere](#).

- Soluzione .NET consolidata per C#: il server C# SDK 5.1+ supporta .NET Framework 4.6.2 (aggiornato dalla versione 4.6.1) e .NET 6.0 in un'unica soluzione. .NET Standard 2.1 è disponibile con le librerie create da Unity.
- Nuova Compute risorsa: questa nuova risorsa combina diversi tipi di risorse di hosting. Include risorse di hosting basate sul cloud (flotte gestite EC2 o container) e risorse di hosting controllate dal cliente (flotte Anywhere). Include i seguenti aggiornamenti:
 - Le nuove chiamate API per la Compute risorsa includono: [ListCompute\(\)](#), [DescribeCompute\(\)](#) e [GetComputeAccess\(\)](#). Queste azioni restituiscono informazioni sulle risorse di hosting per qualsiasi tipo di Amazon GameLift Servers flotta. In generale, per le flotte con server di gioco che utilizzano il server SDK 5.x, utilizza le azioni specifiche del computer per sostituire le azioni specifiche dell'istanza. [Inoltre, queste azioni possono essere utilizzate nelle flotte Anywhere senza l'Amazon GameLift Servers agente: RegisterCompute\(\), \(\) e \(\). DeregisterCompute GetComputeAuthToken](#)
 - Nuova metrica ActiveCompute con CloudWatch dimensioni FleetIdLocation, e. ComputeType Questa metrica sostituisce la metrica precedente. ActiveInstances
- Amazon EC2 Systems Manager (SSM) per l'accesso remoto: per una maggiore sicurezza, utilizza SSM anziché SSH per la connessione a istanze in flotte gestite. Amazon GameLift Servers Consultare [Connect a flotte di istanze](#).

Esegui la migrazione al server SDK 5.x per Amazon GameLift Servers

Per aggiornare un progetto di gioco per utilizzare la versione 5.x del server SDK, apporta le seguenti modifiche:

Note

Se devi continuare a utilizzare la versione 4.x o precedente dell'SDK del server, consulta la documentazione e le informazioni sul [Server SDK per la Amazon GameLift Servers versione 4 e precedenti](#) download. [In particolare, per la versione SDK 4.0.2, puoi scaricarla dalle versioni ufficiali. GitHub](#)

1. Scarica l'SDK del server più recente per il Amazon GameLift Servers pacchetto per il tuo ambiente di sviluppo [[Scarica](#) il sito]. Segui le istruzioni di installazione contenute nel Readme file per il pacchetto e la versione scaricati. Consulta queste istruzioni per utilizzare il server SDKs con il tuo progetto di gioco.

- [Per ambienti di sviluppo che utilizzano C++, C# o Go](#)
- [Per progetti Unreal Engine \(SDK del server C++ solo per le librerie Unreal\)](#)
- [Per i progetti Unity \(SDK del server C# solo per le librerie Unity\)](#)
- [Da utilizzare con il Amazon GameLift Servers plugin per Unreal Engine](#)
- [Da usare con il Amazon GameLift Servers plugin per Unity](#)

2. Aggiorna il codice del server come segue:

- Cambia la funzione di callback del codice server `onCreateGameSession()` in `onStartGameSession()`.
- Aggiorna gli `InitSDK()` input in modo appropriato:
 - Se hai intenzione di distribuire il server di gioco, crea una flotta EC2 Amazon GameLift Servers gestita o una flotta Anywhere con l'agente: Amazon GameLift Servers

[Chiama `InitSDK\(\)` senza parametri \(C++\) \(C#\) \(Unreal\)](#). Questa chiamata configura l'ambiente di calcolo e una WebSocket connessione al servizio. Amazon GameLift Servers

- Se hai intenzione di utilizzare il server di gioco, crea una flotta Anywhere senza l'Amazon GameLift Servers agente:

[Chiama `InitSDK\(\)` con i parametri del server \(C++\) \(C#\) \(Unreal\)](#). Un processo del server di gioco utilizza questi parametri per stabilire una connessione con il servizio. Amazon GameLift Servers

3. Se la build del server di gioco o altre applicazioni ospitate comunicano con altre AWS risorse durante l'esecuzione, dovrai modificare il modo in cui l'applicazione accede a tali risorse. Sostituisci l'uso di `AssumeRoleCredentials` con il nuovo server SDK action `GetFleetRoleCredentials()` (per i server di gioco) o usa credenziali condivise (per altre applicazioni). Per ulteriori informazioni su come implementare questa modifica, consulta. [Connect il server di gioco Amazon GameLift Servers ospitato ad altre AWS risorse](#)

4. [Se il progetto ha chiamato l'azione SDK del server `GetInstanceCertificate\(\)` per recuperare un certificato TLS, modifica il codice per utilizzare invece il nuovo `GetComputeCertificate\(\)` \(C++\) \(C#\) \(Unreal\)](#).

5. Quando carichi la build del gioco su Amazon GameLift Servers (ad esempio con [upload-build](#) o [CreateBuild\(\)](#)), imposta il `ServerSdkVersion` parametro sulla versione 5.x che stai utilizzando (questo parametro attualmente è predefinito su 4.0.2). Questo parametro deve corrispondere alle librerie SDK del server effettive nella build del server di gioco. Se specifichi la versione sbagliata

per una build del server di gioco caricata, tutte le flotte create con quella build falliranno. Per informazioni, consulta [Crea un server di gioco costruito per Amazon GameLift Servers](#).

L'esempio seguente illustra come specificare la versione SDK del server:

```
aws gamelift upload-build \  
  --operating-system AMAZON_LINUX_2023 \  
  --server-sdk-version "5.0.0" \  
  --build-root "~/mygame" \  
  --name "My Game Nightly Build" \  
  --build-version "build 255" \  
  --region us-west-2
```

6. Se utilizzi script per connetterti in remoto a flotte gestite, aggiorna gli script per utilizzare il nuovo processo, come descritto in. [Connect a flotte di istanze](#)

Server C++ SDK 5.x per -- Azioni Amazon GameLift Servers

Usa il riferimento all'SDK 5.x del server per integrare il tuo gioco multiplayer con cui ospitarlo.

Amazon GameLift Servers Per indicazioni sul processo di integrazione, consulta. [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#)

Note

Questo argomento descrive l'API Amazon GameLift Servers C++ che è possibile utilizzare quando si crea con la libreria standard C++ (`std`). In particolare, questa documentazione si applica al codice compilato con l'opzione. `-DDGAMELIFT_USE_STD=1`

Server C++ SDK 5.x per Amazon GameLift Servers -- Tipi di dati

Usa il riferimento all'SDK 5.x del server Amazon GameLift Servers C++ per integrare il tuo gioco multiplayer con cui ospitare. Amazon GameLift Servers Per indicazioni sul processo di integrazione, consulta. [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#)

 Note

Questo argomento descrive l'API Amazon GameLift Servers C++ che è possibile utilizzare quando si crea con la libreria standard C++ (`std`). In particolare, questa documentazione si applica al codice compilato con l'opzione. `-DDGAMELIFT_USE_STD=1`

Server C++ SDK 5.x per -- Azioni Amazon GameLift Servers

Tipi di dati

- [LogParameters](#)
- [MetricsParameters](#)
- [ProcessParameters](#)
- [UpdateGameSession](#)
- [GameSession](#)
- [ServerParameters](#)
- [StartMatchBackfillRequest](#)
- [Player](#)
- [DescribePlayerSessionsRequest](#)
- [StopMatchBackfillRequest](#)
- [AttributeValue](#)
- [GetFleetRoleCredentialsRequest](#)
- [AwsLongOutcome](#)
- [AwsStringOutcome](#)
- [DescribePlayerSessionsOutcome](#)
- [DescribePlayerSessionsResult](#)
- [GenericOutcome](#)
- [GenericOutcomeCallable](#)
- [PlayerSession](#)
- [StartMatchBackfillOutcome](#)
- [StartMatchBackfillResult](#)

- [GetComputeCertificateOutcome](#)
- [GetComputeCertificateResult](#)
- [GetFleetRoleCredentialsOutcome](#)
- [GetFleetRoleCredentialsResult](#)
- [Inizialo SDKOutcome](#)
- [GameLiftError](#)
- [enumerazioni;](#)

LogParameters

Un oggetto che identifica i file generati durante una sessione di gioco che desideri caricare e archiviare Amazon GameLift Servers al termine della sessione di gioco. Il server di gioco LogParameters lo fornisce Amazon GameLift Servers come parte di un ProcessParameters oggetto in una [ProcessReady\(\)](#) chiamata.

Proprietà	Descrizione
LogPaths	L'elenco dei percorsi di directory dei file di registro del server di gioco che desideri Amazon GameLift Servers archiviare per accessi futuri. Il processo del server genera questi file durante ogni sessione di gioco. Definisci i percorsi e i nomi dei file nel tuo server di gioco e li memorizzi nella directory di build del gioco principale. I percorsi di registro devono essere assoluti. Ad esempio, se la build del gioco memorizza i log delle sessioni di gioco in un percorso simile <code>MyGame\sessionLogs\</code>, il percorso si troverebbe <code>c:\game\MyGame\sessionLogs</code> su un'istanza di Windows. Tipo: <code>std::vector<std::string></code> Obbligatorio: no

MetricsParameters

Usa questo tipo di dati per configurare la raccolta delle metriche e la segnalazione degli arresti anomali per il server di gioco. Il server di gioco comunica con `MetricsParameters` durante una `Amazon GameLift Servers InitMetrics\(\)` chiamata.

Proprietà	Descrizione
StatsdHost	Il nome host o l'indirizzo IP del server StatSD per la raccolta delle metriche. Tipo: <code>std::string</code> Obbligatorio: no
StatsdPort	Il numero di porta del server StatSD per la raccolta delle metriche. Tipo: <code>int</code> Obbligatorio: no
CrashReporterHost	Il nome host o l'indirizzo IP del server crash reporter. Tipo: <code>std::string</code> Obbligatorio: no
CrashReporterPort	Il numero di porta del server crash reporter. Tipo: <code>int</code> Obbligatorio: no
FlushIntervalMs	L'intervallo in millisecondi per il trasferimento dei dati delle metriche al server. Tipo: <code>int</code> Obbligatorio: no

MaxPacketSize

La dimensione massima in byte per i pacchetti di metriche inviati al server.

Tipo: `int`

Obbligatorio: no

ProcessParameters

Questo tipo di dati contiene l'insieme di parametri inviati a `Amazon GameLift Servers`.

[ProcessReady\(\)](#)

Proprietà	Descrizione
LogParameters	Un oggetto con percorsi di directory ai file generati durante una sessione di gioco. Amazon GameLift Servers copia e archivia i file per accessi futuri. Tipo: <code>Aws::GameLift::Server::</code> <u>LogParameters</u> Obbligatorio: no
OnHealthCheck	La funzione di callback che Amazon GameLift Servers richiama per richiedere un rapporto sullo stato di salute del processo del server. Amazon GameLift Servers chiama questa funzione ogni 60 secondi e attende una risposta per 60 secondi. Il processo del server ritorna <code>TRUE</code> se integro, <code>FALSE</code> se non integro. Se non viene restituita alcuna risposta, Amazon GameLift Servers registra il processo del server come non integro. Tipo: <code>std::function<bool()> onHealthCheck</code>

Obbligatorio: no

OnProcessTerminate

La funzione di callback che Amazon GameLift Servers richiama per forzare l'arresto del processo server. Dopo aver chiamato questa funzione, Amazon GameLift Servers attende 5 minuti che il processo server si spenga e risponda con una [ProcessEnding\(\)](#) chiamata prima di chiudere il processo server.

Tipo: `std::function<void()>`
`onProcessTerminate`

Obbligatorio: sì

OnStartGameSession

La funzione di callback che Amazon GameLift Servers richiama l'attivazione di una nuova sessione di gioco. Amazon GameLift Servers chiama questa funzione in risposta a una richiesta del client. [CreateGameSession](#) La funzione di callback passa un [GameSession](#) oggetto.

Tipo: `const std::function<void (Aws::GameLift::Model::GameSession)>` `onStartGameSession`

Obbligatorio: sì

OnUpdateGameSession	La funzione di callback che Amazon GameLift Servers richiama per passare un oggetto di sessione di gioco aggiornato al processo del server. Amazon GameLift Servers chiama questa funzione quando è stata elaborata una richiesta di match backfill per fornire dati aggiornati sul matchmaker. Passa un GameSession oggetto, uno status update (updateReason) e l'ID del ticket match backfill. Tipo: <code>std::function<void(Aws::GameLift::Server::Model::UpdateGameSession)> onUpdateGameSession</code> Obbligatorio: no
Porta	Il numero di porta su cui il processo server ascolta le connessioni di nuovi giocatori . Il valore deve rientrare nella gamma di porte configurate per qualsiasi parco istanze che distribuisce questa build del server di gioco. Questo numero di porta è incluso nella sessione di gioco e negli oggetti della sessione del giocatore utilizzati dalle sessioni di gioco per la connessione a un processo server. Tipo: Integer Obbligatorio: sì

UpdateGameSession

Questo tipo di dati viene aggiornato a un oggetto della sessione di gioco, che include il motivo per cui la sessione di gioco è stata aggiornata e il relativo ID del ticket di backfill se il backfill viene utilizzato per riempire le sessioni dei giocatori nella sessione di gioco.

Properties	Descrizione
GameSession	Oggetto GameSession. L'GameSession oggetto contiene proprietà che descrivono una sessione di gioco. Tipo: <code>Aws::GameLift::Server::GameSession</code> Obbligatorio: sì
UpdateReason	Il motivo per cui la sessione di gioco viene aggiornata. Tipo: <code>Aws::GameLift::Server::UpdateReason</code> Obbligatorio: sì
BackfillTicketId	L'ID del ticket di backfill che tenta di aggiornare la sessione di gioco. Tipo: <code>std::string</code> Obbligatorio: no

GameSession

Questo tipo di dati fornisce i dettagli di una sessione di gioco.

Properties	Descrizione
GameSessionId	Un identificatore univoco per la sessione di gioco. L'ARN di una sessione di gioco ha il seguente formato: <code>arn:aws:gamelift:<region>::gamesession/<fleet ID>/<custom ID string or idempotency token></code>

Properties	Descrizione
	Tipo: <code>std::string</code> Obbligatorio: no
Name	Un'etichetta descrittiva della sessione di gioco. Tipo: <code>std::string</code> Obbligatorio: no
FleetId	Un identificatore univoco per la flotta su cui è in esecuzione la sessione di gioco. Tipo: <code>std::string</code> Obbligatorio: no
MaximumPlayerSessionCount	Il numero massimo di connessioni dei giocatori alla sessione di gioco. Tipo: <code>int</code> Obbligatorio: no
Porta	Il numero di porta per la sessione di gioco. Per connettersi a un server di Amazon GameLift Servers gioco, un'app necessita sia dell'indirizzo IP che del numero di porta. Tipo: <code>int</code> Obbligatorio: no

Properties	Descrizione
IpAddress	L'indirizzo IP della sessione di gioco. Per connettersi a un server di Amazon GameLift Servers gioco, un'app necessita sia dell'indirizzo IP che del numero di porta. Tipo: <code>std::string</code> Obbligatorio: no
GameSessionData	Un set di proprietà personalizzate della sessione di gioco, formattate come valore di stringa singola. Tipo: <code>std::string</code> Obbligatorio: no
MatchmakerData	Informazioni sul processo di matchmaking utilizzato per creare la sessione di gioco, in sintassi JSON, formattate come stringa. Oltre alla configurazione di matchmaking utilizzata, contiene dati su tutti i giocatori assegnati alla partita, compresi gli attributi dei giocatori e le assegnazioni delle squadre. Tipo: <code>std::string</code> Obbligatorio: no
GameProperties	Un insieme di proprietà personalizzate per una sessione di gioco, formattate come coppie chiave:valore. Queste proprietà vengono passate con una richiesta di avvio di una nuova sessione di gioco. Tipo: <code>std::vector < GameProperty ></code> Obbligatorio: no

Properties	Descrizione
DnsName	L'identificatore DNS assegnato all'istanza che esegue la sessione di gioco. I valori hanno il seguente formato: • Flotte abilitate per TLS: <code><unique identifier>.<region identifier>.amazongamelift.com</code> • Non-TLS-enabled flotte: <code>ec2-<unique identifier>.compute.amazonaws.com</code> Quando ti connetti a una sessione di gioco in esecuzione su una flotta abilitata per TLS, devi usare il nome DNS, non l'indirizzo IP. Tipo: <code>std::string</code> Obbligatorio: no

ServerParameters

Informazioni utilizzate da un processo del server di gioco per stabilire una connessione con il servizio. Amazon GameLift Servers Include questi parametri durante la chiamata [InitSDK\(\)](#) solo se la build del server di gioco verrà distribuita su una flotta Anywhere o su una flotta di container senza l'Amazon GameLift Servers agente. Per tutti gli altri scenari di implementazione, chiama [InitSDK\(\)](#) senza parametri.

Properties	Descrizione
websocketUrl	Il <code>GameLiftServerSdkEndpoint</code> Amazon GameLift Servers restituisce quando si utilizza RegisterCompute una risorsa di calcolo Amazon GameLift Servers Anywhere. Tipo: <code>std::string</code>

Properties	Descrizione
	Obbligatorio: sì
ID del processo	Un identificatore univoco registrato nel processo del server che ospita il gioco. Tipo: <code>std::string</code> Obbligatorio: sì
hostId	HostIDViene ComputeName utilizzato al momento della registrazione del computer. Per ulteriori informazioni, consultare RegisterCompute. Tipo: <code>std::string</code> Obbligatorio: sì
FleetID	L'identificatore univoco del parco macchine su cui è registrato il computer. Per ulteriori informazioni, consultare RegisterCompute. Tipo: <code>std::string</code> Obbligatorio: sì
AuthToken	Il token di autenticazione generato da Amazon GameLift Servers questo autentica il server su. Amazon GameLift Servers Per ulteriori informazioni, consultare GetComputeAuthToken. Tipo: <code>std::string</code> Obbligatorio: sì

StartMatchBackfillRequest

Informazioni utilizzate per creare una richiesta di matchmaking backfill. Il server di gioco comunica queste informazioni durante una chiamata Amazon GameLift Servers. [StartMatchBackfill\(\)](#)

Properties	Descrizione
GameSessionArn	Un identificatore univoco della sessione di gioco. L'operazione API GetGameSessionId restituisce l'identificatore in formato ARN. Tipo: <code>std::string</code> Obbligatorio: sì
MatchmakingConfigurationArn	Un identificatore univoco, sotto forma di ARN, che il matchmaker può utilizzare per questa richiesta. L'ARN del matchmaker per la sessione di gioco originale si trova nell'oggetto della sessione di gioco nella proprietà matchmaker data. Scopri di più sui dati dei matchmaker in Work with matchmaker data. Tipo: <code>std::string</code> Obbligatorio: sì
Players	Un insieme di dati che rappresenta tutti i giocatori che partecipano alla sessione di gioco. Il matchmaker utilizza queste informazioni per cercare nuovi giocatori che rappresentino un buon abbinamento per i giocatori attuali. Tipo: <code>std::vector<Player></code> Obbligatorio: sì
TicketId	Un identificatore univoco per un ticket di richiesta di matchmaking o match backfill.

Properties	Descrizione
	Se non fornisci un valore, Amazon GameLift Servers ne genera uno. Utilizzare questo identificatore per monitorare lo stato del ticket di backfill degli abbinamenti o annullare la richiesta, se necessario. Tipo: <code>std::string</code> Obbligatorio: no

Player

Questo tipo di dati rappresenta un giocatore in matchmaking. Quando si avvia una richiesta di matchmaking, un giocatore ha un ID giocatore, attributi e possibilmente dati di latenza. Amazon GameLift Servers aggiunge le informazioni sulla squadra dopo una partita.

Properties	Descrizione
LatencyInSM	Un insieme di valori espressi in millisecondi che indica la quantità di latenza che un giocatore sperimenta quando è connesso a una posizione. Se viene utilizzata questa proprietà, il giocatore viene abbinato solo per le posizioni elencate. Se un matchmaker ha una regola che valuta la latenza dei giocatori, i giocatori devono segnalare la latenza per essere abbinati. Tipo: <code>Dictionary<string,int></code> Obbligatorio: no
PlayerAttributes	Una raccolta di coppie chiave:valore contenenti informazioni sui giocatori da utilizzare nel matchmaking. Le chiavi degli attributi del giocatore devono corrispondere a quelle

Properties	Descrizione
	PlayerAttributes utilizzate in un set di regole di matchmaking. Per ulteriori informazioni sugli attributi dei giocatori, vedi AttributeValue. Tipo: <code>std::map<std::string, AttributeValue></code> Obbligatorio: no
PlayerId	Un identificatore univoco per un giocatore. Tipo: <code>std::string</code> Obbligatorio: no
Team	Il nome della squadra a cui è assegnato il giocatore in una partita. Definisci il nome della squadra nel set di regole del matchmaking. Tipo: <code>std::string</code> Obbligatorio: no

DescribePlayerSessionsRequest

Un oggetto che specifica le sessioni dei giocatori da recuperare. Il processo server fornisce queste informazioni con una [DescribePlayerSessions\(\)](#) chiamata a. Amazon GameLift Servers

Properties	Descrizione
GameSessionId	Un identificatore univoco della sessione di gioco. Utilizzare questo parametro per richiedere e tutte le sessioni giocatore per la sessione di gioco specificata.

Properties	Descrizione
	Il formato dell'ID della sessione di gioco è <code>arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string></code>. <code>GameSessionID</code> È una stringa ID personalizzata o un Tipo: <code>std::string</code> Obbligatorio: no
PlayerSessionId	L'identificatore univoco per una sessione di gioco. Usa questo parametro per richiedere una singola sessione di gioco specifica. Tipo: <code>std::string</code> Obbligatorio: no
PlayerId	L'identificatore univoco di un giocatore. Usa questo parametro per richiedere tutte le sessioni di gioco per un giocatore specifico. Per informazioni, consulta Genera giocatore IDs. Tipo: <code>std::string</code> Obbligatorio: no

Properties	Descrizione
PlayerSessionStatusFilter	Lo stato della sessione del giocatore su cui filtrare i risultati. I possibili stati delle sessioni di gioco includono: • RISERVATO — La richiesta di sessione del giocatore è stata ricevuta, ma il giocatore non si è connesso al processo del server né è stato convalidato. • ATTIVO: il giocatore è stato convalidato dal processo del server ed è connesso. • COMPLETATA — La connessione del giocatore è stata interrotta. • TIMEDOUT: è stata ricevuta una richiesta di sessione, ma il giocatore non si è connesso o non è stato convalidato entro il limite di timeout (60 secondi). Tipo: <code>std::string</code> Obbligatorio: no
NextToken	Il token che indica l'inizio della pagina successiva di risultati. Per specificare l'inizio del set di risultati, non fornire un valore. Se fornisci un ID di sessione del giocatore, questo parametro viene ignorato. Tipo: <code>std::string</code> Obbligatorio: no

Properties	Descrizione
Limite	Numero massimo di risultati da restituire. Se fornisci un ID di sessione del giocatore, questo parametro viene ignorato. Tipo: <code>int</code> Obbligatorio: no

StopMatchBackfillRequest

Informazioni utilizzate per annullare una richiesta di matchmaking backfill. Il server di gioco comunica queste informazioni al Amazon GameLift Servers servizio durante una chiamata. [StopMatchBackfill\(\)](#)

Properties	Descrizione
GameSessionArn	Un identificatore univoco della sessione di gioco della richiesta annullata. Tipo: <code>char[]</code> Obbligatorio: no
MatchmakingConfigurationArn	Un identificatore univoco del matchmaker a cui è stata inviata questa richiesta. Tipo: <code>char[]</code> Obbligatorio: no
TicketId	Un identificatore univoco del ticket di richiesta di riempimento da annullare. Tipo: <code>char[]</code> Obbligatorio: no

AttributeValue

Utilizzate questi valori nelle [Player](#) coppie chiave-valore degli attributi. Questo oggetto consente di specificare il valore di un attributo utilizzando uno qualsiasi dei tipi di dati validi: stringa, numero, array di stringhe o mappa di dati. Ogni `AttributeValue` oggetto deve utilizzare esattamente una delle proprietà disponibili: SN,SL, oSDM.

Properties	Description
AttrType	Speciifica il tipo di valore dell'attributo. I possibili tipi di valori degli attributi includono: • NONE • STRINGA • DOPPIA • ELENCO_STRINGHE • STRING_DOUBLE_MAP Obbligatorio: no
S	Rappresenta un valore di attributo di tipo stringa. Tipo: <code>std::string</code> Obbligatorio: no
N	Rappresenta un valore di attributo numerico. Tipo: <code>double</code> Obbligatorio: no
SL	Rappresenta una matrice di valori di attributi di tipo stringa. Tipo: <code>std::vector<std::string></code> Obbligatorio: no
SDM	Rappresenta un dizionario di chiavi di stringa e valori doppi.

Properties	Description
	Tipo: <code>std::map<std::string, double></code> Obbligatorio: no

GetFleetRoleCredentialsRequest

Questo tipo di dati offre al server di gioco un accesso limitato alle altre risorse. AWS Per ulteriori informazioni, consultare [Configurare un ruolo di servizio IAM per Amazon GameLift Servers](#).

Properties	Descrizione
RoleArn	L'Amazon Resource Name (ARN) del ruolo di servizio che estende l'accesso limitato alle tue AWS risorse. Tipo: <code>std::string</code> Obbligatorio: no
RoleSessionName	Il nome della sessione di ruolo che puoi utilizzare e per identificare in modo univoco una AWS Security Token Service AssumeRole sessione. Questo nome è esposto nei log di controllo, come quelli di CloudTrail Tipo: <code>std::string</code> Obbligatorio: no

AwsLongOutcome

Questo tipo di dati risulta da un'azione e produce un oggetto con le seguenti proprietà:

Properties	Description
Risultato	Il risultato dell'azione.

Properties	Description
	Tipo: long Obbligatorio: no
ResultWithOwnership	Il risultato dell'azione, espresso come rvalue, in modo che il codice chiamante possa assumere la proprietà dell'oggetto. Tipo: long&& Obbligatorio: no
Completato	Se l'azione ha avuto successo o meno. Tipo: bool Obbligatorio: sì
Errore	L'errore che si è verificato se l'azione non ha avuto successo. Tipo: the section called “GameLiftError” Obbligatorio: no

AwsStringOutcome

Questo tipo di dati risulta da un'azione e produce un oggetto con le seguenti proprietà:

Properties	Description
Risultato	Il risultato dell'azione. Tipo: std::string Obbligatorio: no
ResultWithOwnership	Il risultato dell'azione, espresso come rvalue, in modo che il codice chiamante possa assumere la proprietà dell'oggetto. Tipo: long&&

Properties	Description
	Obbligatorio: no
Completato	Se l'azione ha avuto successo o meno. Tipo: bool Obbligatorio: sì
Errore	L'errore che si è verificato se l'azione non ha avuto successo. Tipo: the section called “GameLiftError” Obbligatorio: no

DescribePlayerSessionsOutcome

Questo tipo di dati risulta da un'azione e produce un oggetto con le seguenti proprietà:

Properties	Description
Risultato	Il risultato dell'azione. Tipo: the section called “DescribePlayerSessionsResult” Obbligatorio: no
ResultWithOwnership	Il risultato dell'azione, espresso come rvalue, in modo che il codice chiamante possa assumere la proprietà dell'oggetto. Tipo: <code>Aws::GameLift::Server::Model::DescribePlayerSessionsResult&&</code> Obbligatorio: no
Completato	Se l'azione ha avuto successo o meno. Tipo: bool Obbligatorio: sì

Properties	Description
Errore	L'errore che si è verificato se l'azione non ha avuto successo. Tipo: the section called "GameLiftError" Obbligatorio: no

DescribePlayerSessionsResult

Una raccolta di oggetti contenente proprietà per ogni sessione di giocatore che corrisponde alla richiesta.

Properties	Description
NextToken	Un token che indica l'inizio della successiva pagina sequenziale di risultati. Utilizzate il token restituito con una precedente chiamata a questa operazione. Per iniziare dall'inizio del set di risultati, non specificate un valore. Se viene specificato un ID sessione giocatore, questo parametro verrà ignorato. Tipo: <code>std::string</code> Obbligatorio: sì
PlayerSessions	Tipo: <code>IList<the section called "PlayerSession"></code> Campo obbligatorio:
ResultWithOwnership	Il risultato dell'azione, espresso come rvalue, in modo che il codice chiamante possa assumere la proprietà dell'oggetto. Tipo: <code>std::string&&</code> Obbligatorio: no
Completato	Se l'azione ha avuto successo o meno. Tipo: <code>bool</code>

Properties	Description
	Obbligatorio: sì
Errore	L'errore che si è verificato se l'azione non ha avuto successo. Tipo: the section called “GameLiftError” Obbligatorio: no

GenericOutcome

Questo tipo di dati risulta da un'azione e produce un oggetto con le seguenti proprietà:

Properties	Description
Completato	Se l'azione ha avuto successo o meno. Tipo: bool Obbligatorio: sì
Errore	L'errore che si è verificato se l'azione non ha avuto successo. Tipo: the section called “GameLiftError” Obbligatorio: no

GenericOutcomeCallable

Questo tipo di dati è un risultato generico asincrono. Possiede le seguenti proprietà:

Properties	Description
Completato	Se l'azione ha avuto successo o meno. Tipo: bool Obbligatorio: sì

Properties	Description
Errore	L'errore che si è verificato se l'azione non ha avuto successo. Tipo: the section called "GameLiftError" Obbligatorio: no

PlayerSession

Questo tipo di dati rappresenta una sessione del giocatore che passa al server di gioco. Amazon GameLift Servers Per ulteriori informazioni, consulta [PlayerSession](#).

Properties	Description
CreationTime	Tipo: long Obbligatorio: no
FleetId	Tipo: std::string Obbligatorio: no
GameSessionId	Tipo: std::string Obbligatorio: no
IpAddress	Tipo: std::string Obbligatorio: no
PlayerData	Tipo: std::string Obbligatorio: no
PlayerId	Tipo: std::string Obbligatorio: no
PlayerSessionId	Tipo: std::string

Properties	Description
	Obbligatorio: no
Porta	Tipo: <code>int</code> Obbligatorio: no
Status	Stato sessione giocatore su cui filtrare i risultati. Quando <code>PlayerId</code> viene fornito un <code>PlayerSessionId</code> or, non <code>PlayerSessionStatusFilter</code> ha alcun effetto sulla risposta. Tipo: Un <code>PlayerSessionStatus</code> enum. I valori possibili sono: • ACTIVE • COMPLETED • NOT_SET • RISERVATO • TIMEOUT Obbligatorio: no
TerminationTime	Tipo: <code>long</code> Obbligatorio: no
DnsName	Tipo: <code>std::string</code> Obbligatorio: no

StartMatchBackfillOutcome

Questo tipo di dati risulta da un'azione e produce un oggetto con le seguenti proprietà:

Properties	Description
Risultato	Il risultato dell'azione.

Properties	Description
	Tipo: the section called “StartMatchBackfillResult” Obbligatorio: no
ResultWithOwnership	Il risultato dell'azione, espresso come rvalue, in modo che il codice chiamante possa assumere la proprietà dell'oggetto. Tipo: <code>StartMatchBackfillResult&&</code> Obbligatorio: no
Completato	Se l'azione ha avuto successo o meno. Tipo: <code>bool</code> Obbligatorio: sì
Errore	L'errore che si è verificato se l'azione non ha avuto successo. Tipo: the section called “GameLiftError” Obbligatorio: no

StartMatchBackfillResult

Questo tipo di dati risulta da un'azione e produce un oggetto con le seguenti proprietà:

Properties	Description
TicketId	Un identificatore univoco per un ticket di matchmaking. Se qui non viene specificato alcun ID del ticket, ne Amazon GameLift Servers genererà uno sotto forma di UUID. Usa questo identificatore per tenere traccia dello stato del ticket di backfill della partita e recuperare i risultati della partita. Tipo: <code>std::string</code> Obbligatorio: no

GetComputeCertificateOutcome

Questo tipo di dati risulta da un'azione e produce un oggetto con le seguenti proprietà:

Properties	Description
Risultato	Il risultato dell'azione. Tipo: the section called “GetComputeCertificateResult” Obbligatorio: no
ResultWithOwnership	Il risultato dell'azione, espresso come rvalue, in modo che il codice chiamante possa assumere la proprietà dell'oggetto. Tipo: <code>Aws::GameLift::Server::Model::GetComputeCertificateResult&&</code> Obbligatorio: no
Completato	Se l'azione ha avuto successo o meno. Tipo: <code>bool</code> Obbligatorio: sì
Errore	L'errore che si è verificato se l'azione non ha avuto successo. Tipo: the section called “GameLiftError” Obbligatorio: no

GetComputeCertificateResult

Il percorso del certificato TLS sul computer e il nome host del computer.

Properties	Description
CertificatePath	Il percorso del certificato TLS sulla tua risorsa di elaborazione. Quando si utilizza una flotta Amazon GameLift Servers gestita, questo percorso contiene:

Properties	Description
	• <code>certificate.pem</code> : Il certificato dell'utente finale. La catena completa di certificati è la combinazione dei certificati <code>certificateChain.pem</code> aggiunti a questo certificato. • <code>certificateChain.pem</code> : La catena di certificati che contiene il certificato principale e i certificati intermedi. • <code>rootCertificate.pem</code> : Il certificato principale. • <code>privateKey.pem</code> : la chiave privata per il certificato dell'utente finale. Tipo: <code>std::string</code> Obbligatorio: no
ComputeName	Il nome della risorsa di calcolo. Tipo: <code>std::string</code> Obbligatorio: no

GetFleetRoleCredentialsOutcome

Questo tipo di dati risulta da un'azione e produce un oggetto con le seguenti proprietà:

Properties	Description
Risultato	Il risultato dell'azione. Tipo: the section called “GetFleetRoleCredentialsResult” Obbligatorio: no
ResultWithOwnership	Il risultato dell'azione, espresso come rvalue, in modo che il codice chiamante possa assumere la proprietà dell'oggetto. Tipo: <code>Aws::GameLift::Server::Model::GetFleetRoleCredentialsResult</code>

Properties	Description
	Obbligatorio: no
Completato	Se l'azione ha avuto successo o meno. Tipo: <code>bool</code> Obbligatorio: sì
Errore	L'errore che si è verificato se l'azione non ha avuto successo. Tipo: the section called "GameLiftError" Obbligatorio: no

GetFleetRoleCredentialsResult

Properties	Description
AccessKeyId	L'ID della chiave di accesso per autenticare e fornire l'accesso alle tue risorse. AWS Tipo: <code>string</code> Obbligatorio: no
AssumedRoleId	L'ID dell'utente a cui appartiene il ruolo di servizio. Tipo: <code>string</code> Obbligatorio: no
AssumedRoleUserArn	L'Amazon Resource Name (ARN) dell'utente a cui appartiene il ruolo di servizio. Tipo: <code>string</code> Obbligatorio: no

Properties	Description
Scadenza	Il periodo di tempo che manca alla scadenza delle credenziali di sessione. Tipo: <code>DateTime</code> Obbligatorio: no
SecretAccessKey	L'ID della chiave di accesso segreta per l'autenticazione. Tipo: <code>string</code> Obbligatorio: no
SessionToken	Un token per identificare la sessione attiva corrente che interagisce con AWS le tue risorse. Tipo: <code>string</code> Obbligatorio: no
Completato	Se l'azione ha avuto successo o meno. Tipo: <code>bool</code> Obbligatorio: sì
Errore	L'errore che si è verificato se l'azione non ha avuto successo. Tipo: the section called “GameLiftError” Obbligatorio: no

Inizialo SDKOutcome

 Note

`InitSDKOutcome` viene restituito solo quando si crea l'SDK con il flag. `std` Se si crea con la `nostd` bandiera, allora [the section called “GenericOutcome”](#) viene invece restituito.

Properties	Description
Completato	Se l'azione ha avuto successo o meno. Tipo: bool Obbligatorio: sì
Errore	L'errore che si è verificato se l'azione non ha avuto successo. Tipo: the section called "GameLiftError" Obbligatorio: no

GameLiftError

Properties	Description
ErrorType	Il tipo di errore. Tipo: Un enum. GameLiftErrorType Obbligatorio: no
ErrorMessage	Messaggio di errore. Tipo: std::string Obbligatorio: no
ErrorName	Il nome dell'errore. Tipo: std::string Obbligatorio: no

enumerazioni;

Le enumerazioni definite per l'SDK del server per Amazon GameLift Servers (C++) sono definite come segue:

GameLiftErrorType

Valore di stringa che indica il tipo di errore. I valori validi includono:

- `BAD_REQUEST_EXCEPTION`
- `GAMESESSION_ID_NOT_SET` — L'ID della sessione di gioco non è stato impostato.
- `INTERNAL_SERVICE_EXCEPTION`
- `LOCAL_CONNECTION_FAILED` — La connessione locale a non è riuscita. Amazon GameLift Servers
- `NETWORK_NOT_INITIALIZED` — La rete non è stata inizializzata.
- `SERVICE_CALL_FAILED` — Una chiamata a un servizio non è riuscita. AWS
- `WEBSOCKET_CONNECT_FAILURE`
- `WEBSOCKET_CONNECT_FAILURE_FORBIDDEN`
- `URL_NON_VALIDO DI WEBSOCKET_CONNECT_FAILURE_NON_VALIDO`
- `WEBSOCKET_CONNECT_FAILURE_TIMEOUT`
- `ALREADY_INITIALIZED` — Il server o il client è già stato inizializzato con `Initialize ()`. Amazon GameLift Servers
- `FLEET_MISMATCH` — La flotta bersaglio non corrisponde alla flotta di una `GameSession` o `PlayerSession`.
- `GAMELIFT_CLIENT_NOT_INITIALIZED` — Il client non è stato inizializzato. Amazon GameLift Servers
- `GAMELIFT_SERVER_NOT_INITIALIZED` — Amazon GameLift Servers Il server non è stato inizializzato.
- `GAME_SESSION_ENDED_FAILED`: l'SDK del server non è riuscito a contattare il servizio per segnalare la fine della sessione di gioco. Amazon GameLift Servers
- `GAME_SESSION_NOT_READY` — La sessione di gioco sul server non è stata attivata. Amazon GameLift Servers
- `GAME_SESSION_READY_FAILED`: l'SDK del server per cui non è stato possibile contattare il servizio per segnalare che la sessione di gioco è pronta. Amazon GameLift Servers
- `INITIALIZATION_MISMATCH` — È stato chiamato un metodo client dopo `Server: :Initialize ()` o viceversa.
- `NOT_INITIALIZED` — Il Amazon GameLift Servers server o il client non sono stati inizializzati con `Initialize ()`.

- `NO_TARGET_ALIASID_SET` — Non è stato impostato un aliasID di destinazione.
- `NO_TARGET_FLEET_SET` — Non è stata impostata una flotta di obiettivi.
- `PROCESS_ENDING_FAILED`: l'SDK del server non è riuscito a contattare il servizio per segnalare la fine del processo. Amazon GameLift Servers
- `PROCESS_NOT_ACTIVE` — Il processo del server non è ancora attivo, non è associato a un e non può accettare o elaborare. GameSession PlayerSessions
- `PROCESS_NOT_READY` — Il processo del server non è ancora pronto per essere attivato.
- `PROCESS_READY_FAILED`: l'SDK del server non è riuscito a contattare il servizio per Amazon GameLift Servers segnalare che il processo è pronto.
- `SDK_VERSION_DETECTION_FAILED` — Rilevamento della versione SDK non riuscito.
- `STX_CALL_FAILED` — Una chiamata al componente di backend del server non è riuscita. XStx
- `STX_INITIALIZATION_FAILED` — L'inizializzazione del componente di backend del server non è riuscita. XStx
- `UNEXPECTED_PLAYER_SESSION` — Il server ha rilevato una sessione di giocatore non registrata.
- `WEBSOCKET_CONNECT_FAILURE`
- `WEBSOCKET_CONNECT_FAILURE_FORBIDDEN`
- `URL NON VALIDO DI WEBSOCKET_CONNECT_FAILURE_NON_VALIDO`
- `WEBSOCKET_CONNECT_FAILURE_TIMEOUT`
- `WEBSOCKET_RETRIABLE_SEND_MESSAGE_FAILURE` — Errore recuperabile nell'invio di un messaggio al servizio. GameLift WebSocket
- `WEBSOCKET_SEND_MESSAGE_FAILURE` — Errore nell'invio di un messaggio al GameLift servizio. WebSocket
- `MATCH_BACKFILL_REQUEST_VALIDATION` — La convalida della richiesta non è riuscita.
- `PLAYER_SESSION_REQUEST_VALIDATION` — La convalida della richiesta non è riuscita.

PlayerSessionCreationPolicy

Valore della stringa che indica se la sessione di gioco accetta nuovi giocatori. I valori validi includono:

- `ACCEPT_ALL`: accetta tutte le nuove sessioni giocatore.
- `DENY_ALL`: rifiuta tutte le nuove sessioni giocatore.

- NOT_SET — La sessione di gioco non è impostata per accettare o rifiutare sessioni di nuovi giocatori.

[Server C++ SDK 5.x per Amazon GameLift Servers -- Tipi di dati](#)

Argomenti

- [GetSdkVersion\(\)](#)
- [InitMetrics\(\)](#)
- [InitSDK\(\)](#)
- [InitSDK\(\)](#)
- [ProcessReady\(\)](#)
- [ProcessReadyAsync\(\)](#)
- [ProcessEnding\(\)](#)
- [ActivateGameSession\(\)](#)
- [UpdatePlayerSessionCreationPolicy\(\)](#)
- [GetGameSessionId\(\)](#)
- [GetTerminationTime\(\)](#)
- [AcceptPlayerSession\(\)](#)
- [RemovePlayerSession\(\)](#)
- [DescribePlayerSessions\(\)](#)
- [StartMatchBackfill\(\)](#)
- [StopMatchBackfill\(\)](#)
- [GetComputeCertificate\(\)](#)
- [GetFleetRoleCredentials\(\)](#)
- [Distruggi \(\)](#)

GetSdkVersion()

Restituisce il numero di versione corrente dell'SDK integrato nel processo del server.

Sintassi

```
Aws::GameLift::AwsStringOutcome Server::GetSdkVersion();
```

Valore restituito

Se l'esito è positivo, restituisce la versione corrente dell'SDK come oggetto [the section called “AwsStringOutcome”](#). L'oggetto restituito include il numero di versione (esempio `5.0.0`). Se l'esito è negativo, verrà restituito un messaggio di errore.

Esempio

```
Aws::GameLift::AwsStringOutcome SdkVersionOutcome =  
    Aws::GameLift::Server::GetSdkVersion();
```

InitMetrics()

Inizializza il sistema di metriche per la raccolta e la segnalazione dei dati sulle prestazioni del server. Per ottenere i migliori risultati, chiamate prima questo metodo per abilitare la raccolta delle metriche [InitSDK\(\)](#) durante l'inizializzazione dell'SDK.

Sintassi

```
Aws::GameLift::GenericOutcome InitMetrics();  
Aws::GameLift::GenericOutcome InitMetrics(const  
    Aws::GameLift::Server::MetricsParameters &metricsParameters);
```

Parameters

[MetricsParameters](#) (facoltativo)

Parametri di configurazione per la raccolta delle metriche. Se non viene fornito, utilizza valori predefiniti che possono essere sostituiti dalle variabili di ambiente.

Valore restituito

In caso di successo, restituisce un [the section called “GenericOutcome”](#) oggetto che indica il successo. Se l'esito è negativo, verrà restituito un messaggio di errore.

Esempio

```
// Initialize with default parameters (uses environment variables if available)  
// Defaults: localhost:8125 for StatsD, localhost:8126 for crash reporter  
// FlushInterval: 10000ms, MaxPacketSize: 512 bytes
```

```
Aws::GameLift::GenericOutcome outcome = Aws::GameLift::Server::InitMetrics();
if (outcome.IsSuccess()) {
    // Metrics system initialized successfully
}

// Initialize with custom parameters
Aws::GameLift::Server::MetricsParameters metricsParams("localhost", 8125, "crash-host",
    8126, 5000, 1024);
Aws::GameLift::GenericOutcome customOutcome =
    Aws::GameLift::Server::InitMetrics(metricsParams);
if (customOutcome.IsSuccess()) {
    // Metrics system initialized with custom parameters
}
```

InitSDK()

Inizializza l'SDK Amazon GameLift Servers. Chiama questo metodo all'avvio prima di qualsiasi altro passaggio di inizializzazione relativo a Amazon GameLift Servers. Questa azione legge i parametri del server dall'ambiente host per impostare la comunicazione tra il processo del server di gioco e il Amazon GameLift Servers servizio. Utilizza un token di idempotenza, quindi puoi riprovare a ripetere questa chiamata in tutta sicurezza quando fallisce.

Se la build del server di gioco verrà implementata senza l'Amazon GameLift ServersAgent to a Amazon GameLift Servers Anywhere fleet o la flotta di container, chiama [InitSDK\(\)](#) e specifica una serie di parametri del server.

Sintassi

```
Server::InitSDKOutcome Server::initSdkOutcome = InitSDK();
```

Valore restituito

Restituisce un [the section called “Inizialo SDKOutcome”](#) oggetto che indica se il processo del server è pronto per la chiamata. [ProcessReady\(\)](#)

Esempio

```
//Call InitSDK to establish a local connection with the Amazon GameLift Servers Agent
to enable further communication.
Aws::GameLift::Server::InitSDKOutcome initSdkOutcome =
    Aws::GameLift::Server::InitSDK();
```


InitSDK()

Inizializza l'SDK Amazon GameLift Servers. Chiama questo metodo all'avvio prima di qualsiasi altro passaggio di inizializzazione relativo a Amazon GameLift Servers. Questa azione richiede una serie di parametri del server per impostare la comunicazione tra il processo del server di gioco e il Amazon GameLift Servers servizio. Utilizza un token di idempotenza, quindi puoi riprovare a ripetere questa chiamata in tutta sicurezza quando fallisce.

Se la build del server di gioco verrà distribuita su una flotta EC2 Amazon GameLift Servers gestita o su una flotta o una flotta di container Amazon GameLift Servers Anywhere con l'Amazon GameLift ServersAgent, chiama senza parametri del server. [InitSDK\(\)](#)

Sintassi

```
Server::InitSDKOutcome Server::initSdkOutcome = InitSDK(serverParameters);
```

Parameters

[ServerParameters](#)

Per inizializzare un server di gioco su una flotta Amazon GameLift Servers Anywhere, costruisci un `ServerParameters` oggetto con le seguenti informazioni:

- L'URL WebSocket utilizzato per connettersi al server di gioco.
- L'ID del processo utilizzato per ospitare il server di gioco.
- L'ID del computer che ospita i processi del server di gioco.
- L'ID della Amazon GameLift Servers flotta contenente il tuo computer Amazon GameLift Servers Anywhere.
- Il token di autorizzazione generato dall'Amazon GameLift Serversoperazione.

Valore restituito

Restituisce un [the section called “Inizialo SDKOutcome”](#) oggetto che indica se il processo del server è pronto per la chiamata. [ProcessReady\(\)](#)

Note

Se le chiamate a non `InitSDK()` riescono per le build di gioco distribuite sulle flotte Anywhere, controlla il `ServerSdkVersion` parametro utilizzato durante la creazione della

risorsa di compilazione. È necessario impostare esplicitamente questo valore sulla versione SDK del server in uso. Il valore predefinito per questo parametro è 4.x, che non è compatibile. Per risolvere questo problema, crea una nuova build e distribuiscila in una nuova flotta.

Esempio

Amazon GameLift ServersEsempio: Ovunque

```
//Define the server parameters
std::string websocketUrl = "wss://us-west-1.api.amazongamelift.com";
std::string processId = "PID1234";
std::string fleetId = "arn:aws:gamelift:us-west-1:111122223333:fleet/
fleet-9999ffff-88ee-77dd-66cc-5555bbbb44aa";
std::string hostId = "HardwareAnywhere";
std::string authToken = "1111aaaa-22bb-33cc-44dd-5555eeee66ff";
Aws::GameLift::Server::Model::ServerParameters serverParameters =
    Aws::GameLift::Server::Model::ServerParameters(webSocketUrl, authToken, fleetId,
        hostId, processId);

//Call InitSDK to establish a local connection with the Amazon GameLift Servers Agent
to enable further communication.
Aws::GameLift::Server::InitSDKOutcome initSdkOutcome =
    Aws::GameLift::Server::InitSDK(serverParameters);
```

ProcessReady()

Notifica Amazon GameLift Servers che il processo del server è pronto per ospitare sessioni di gioco. Chiama questo metodo dopo averlo [InitSDK\(\)](#) invocato. Questo metodo deve essere chiamato solo una volta per processo.

Sintassi

```
GenericOutcome ProcessReady(const Aws::GameLift::Server::ProcessParameters
&processParameters);
```

Parameters

processParameters

Un oggetto [ProcessParameters](#) che comunica le informazioni seguenti sul processo del server:

- Nomi dei metodi di callback implementati nel codice del server di gioco richiamato dal Amazon GameLift Servers servizio per comunicare con il processo del server.
- Numero di porta sulla quale è in ascolto il processo del server.
- Percorso verso qualsiasi file specifico di una sessione di gioco che Amazon GameLift Servers deve acquisire e archiviare.

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

Questo esempio illustra la chiamata [ProcessReady\(\)](#) e le implementazioni della funzione delegata.

```
// Set parameters and call ProcessReady
std::string serverLog("serverOut.log");           // Example of a log file written by the
game server
std::vector<std::string> logPaths;
logPaths.push_back(serverLog);
int listenPort = 9339;

Aws::GameLift::Server::ProcessParameters processReadyParameter =
    Aws::GameLift::Server::ProcessParameters(
        std::bind(&Server::onStartGameSession, this, std::placeholders::_1),
        std::bind(&Server::onProcessTerminate, this),
        std::bind(&Server::OnHealthCheck, this),
        std::bind(&Server::OnUpdateGameSession, this),
        listenPort,
        Aws::GameLift::Server::LogParameters(logPaths)
    );

Aws::GameLift::GenericOutcome outcome =
    Aws::GameLift::Server::ProcessReady(processReadyParameter);

// Implement callback functions
void Server::onStartGameSession(Aws::GameLift::Model::GameSession myGameSession)
{
    // game-specific tasks when starting a new game session, such as loading map
    GenericOutcome outcome =
        Aws::GameLift::Server::ActivateGameSession (maxPlayers);
}
```

```
void Server::onProcessTerminate()
{
    // game-specific tasks required to gracefully shut down a game session,
    // such as notifying players, preserving game state data, and other cleanup
    GenericOutcome outcome = Aws::GameLift::Server::ProcessEnding();
}

bool Server::onHealthCheck()
{
    bool health;
    // complete health evaluation within 60 seconds and set health
    return health;
}
```

ProcessReadyAsync()

Notifica al servizio Amazon GameLift Servers che il processo del server è pronto per l'hosting delle sessioni di gioco. Questo metodo dovrebbe essere chiamato dopo che il processo del server è pronto per ospitare una sessione di gioco. I parametri specificano i nomi delle funzioni di callback Amazon GameLift Servers da chiamare in determinate circostanze. Il codice del server di gioco deve implementare queste funzioni.

Questa chiamata è asincrona. Per effettuare una chiamata sincrona, utilizza [ProcessReady\(\)](#). Per ulteriori dettagli, consulta [Inizializza il processo del server](#).

Sintassi

```
GenericOutcomeCallable ProcessReadyAsync(
    const Aws::GameLift::Server::ProcessParameters &processParameters);
```

Parameters

processParameters

Un oggetto [ProcessParameters](#) che comunica le informazioni seguenti sul processo del server:

- Nomi dei metodi di callback implementati nel codice del server di gioco richiamato dal Amazon GameLift Servers servizio per comunicare con il processo del server.
- Numero di porta sulla quale è in ascolto il processo del server.
- Percorso verso qualsiasi file specifico di una sessione di gioco che Amazon GameLift Servers deve acquisire e archiviare.

Obbligatorio: sì

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

```
// Set parameters and call ProcessReady
std::string serverLog("serverOut.log");           // This is an example of a log file
written by the game server
std::vector<std::string> logPaths;
logPaths.push_back(serverLog);
int listenPort = 9339;

Aws::GameLift::Server::ProcessParameters processReadyParameter =
    Aws::GameLift::Server::ProcessParameters(std::bind(&Server::onStartGameSession, this,
std::placeholders::_1),
    std::bind(&Server::onProcessTerminate, this), std::bind(&Server::OnHealthCheck,
this),
    std::bind(&Server::OnUpdateGameSession, this), listenPort,
    Aws::GameLift::Server::LogParameters(logPaths));

Aws::GameLift::GenericOutcomeCallable outcome =
    Aws::GameLift::Server::ProcessReadyAsync(processReadyParameter);

// Implement callback functions
void onStartGameSession(Aws::GameLift::Model::GameSession myGameSession)
{
    // game-specific tasks when starting a new game session, such as loading map
    GenericOutcome outcome = Aws::GameLift::Server::ActivateGameSession (maxPlayers);
}

void onProcessTerminate()
{
    // game-specific tasks required to gracefully shut down a game session,
    // such as notifying players, preserving game state data, and other cleanup
    GenericOutcome outcome = Aws::GameLift::Server::ProcessEnding();
}

bool onHealthCheck()
{
    // perform health evaluation and complete within 60 seconds
```

```
    return health;
}
```

ProcessEnding()

Notifica Amazon GameLift Servers che il processo del server sta terminando. Chiama questo metodo dopo tutte le altre attività di pulizia (inclusa la chiusura della sessione di gioco attiva) e prima di terminare il processo. A seconda del risultato di `ProcessEnding()`, il processo termina con successo (0) o errore (-1) e genera un evento relativo alla flotta. Se il processo termina con un errore, l'evento fleet generato è `SERVER_PROCESS_TERMINATED_UNHEALTHY`.

Sintassi

```
Aws::GameLift::GenericOutcome processEndingOutcome =
    Aws::GameLift::Server::ProcessEnding();
```

Valore restituito

Restituisce un risultato generico costituito da successo o fallimento con un messaggio di errore.

Esempio

Questo esempio chiama `ProcessEnding()` e `Destroy()` prima di terminare il processo del server con un codice di uscita riuscito o di errore.

```
Aws::GameLift::GenericOutcome processEndingOutcome =
    Aws::GameLift::Server::ProcessEnding();
Aws::GameLift::Server::Destroy();

// Exit the process with success or failure
if (processEndingOutcome.IsSuccess()) {
    exit(0);
}
else {
    cout << "ProcessEnding() failed. Error: " <<
        processEndingOutcome.GetError().GetErrorMessage();
    exit(-1);
}
```

ActivateGameSession()

Notifica Amazon GameLift Servers che il processo del server ha attivato una sessione di gioco ed è ora pronto a ricevere le connessioni dei giocatori. Questa azione dovrebbe essere richiamata come parte della funzione di `onStartGameSession()` callback, dopo l'inizializzazione di tutta la sessione di gioco.

Sintassi

```
Aws::GameLift::GenericOutcome activateGameSessionOutcome =  
    Aws::GameLift::Server::ActivateGameSession();
```

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

Questo esempio mostra le `ActivateGameSession()` chiamate come parte della funzione `onStartGameSession()` delegata.

```
void onStartGameSession(Aws::GameLift::Model::GameSession myGameSession)  
{  
    // game-specific tasks when starting a new game session, such as loading map  
    GenericOutcome outcome = Aws::GameLift::Server::ActivateGameSession();  
}
```

UpdatePlayerSessionCreationPolicy()

Aggiorna la capacità della sessione di gioco corrente di accettare nuove sessioni giocatore. Una sessione di gioco può essere configurata per accettare o rifiutare tutte le nuove sessioni giocatore.

Sintassi

```
GenericOutcome  
    UpdatePlayerSessionCreationPolicy(Aws::GameLift::Model::PlayerSessionCreationPolicy  
        newPlayerSessionPolicy);
```

Parameters

playerCreationSessionPolitica

Tipo: valore `PlayerSessionCreationPolicy` [enum](#).

Obbligatorio: sì

Valore restituito

Restituisce un risultato generico costituito da successo o fallimento con un messaggio di errore.

Esempio

Questo esempio definisce la policy di partecipazione alla sessione di gioco corrente per accettare tutti i giocatori.

```
Aws::GameLift::GenericOutcome outcome =  
  
    Aws::GameLift::Server::UpdatePlayerSessionCreationPolicy(Aws::GameLift::Model::PlayerSessionCr
```

GetGameSessionId()

Recupera l'ID della sessione di gioco ospitata dal processo del server attivo.

Per i processi inattivi che non vengono attivati con una sessione di gioco, la chiamata restituisce un [the section called "GameLiftError"](#).

Sintassi

```
AwsStringOutcome GetGameSessionId()
```

Parameters

Questa operazione non prevede parametri.

Valore restituito

Se l'esito è positivo, l'ID della sessione di gioco verrà restituito come oggetto [the section called "AwsStringOutcome"](#). Se l'esito è negativo, verrà restituito un messaggio di errore.

Per i processi inattivi che non vengono attivati con una sessione di gioco, la chiamata restituisce `Success = True` e `GameSessionId = . ""`

Esempio

```
Aws::GameLift::AwsStringOutcome sessionIdOutcome =  
    Aws::GameLift::Server::GetGameSessionId();
```

GetTerminationTime()

Restituisce il tempo di arresto pianificato di un processo del server, se è disponibile un tempo di chiusura. Un processo server interviene dopo aver ricevuto una `onProcessTerminate()` richiamata da Amazon GameLift Servers. Amazon GameLift Servers chiama `onProcessTerminate()` per i seguenti motivi:

- Quando il processo del server ha segnalato problemi di salute o non ha risposto. Amazon GameLift Servers
- Quando si termina l'istanza durante un evento di scale-down.
- [Quando un'istanza viene terminata a causa di un'interruzione di un'istanza spot.](#)

Sintassi

```
AwsDateTimeOutcome GetTerminationTime()
```

Valore restituito

In caso di successo, restituisce l'ora di terminazione come oggetto. `AwsDateTimeOutcome` Il valore è il tempo di terminazione, espresso in segni di spunta trascorsi da allora. `0001 00:00:00` Ad esempio, il valore della data e dell'ora è uguale ai segni di `2020-09-13 12:26:40 -000Z` spunta. `6373559680000000000` Se non è disponibile alcun orario di terminazione, restituisce un messaggio di errore.

Se il processo non ha ricevuto un `ProcessParameters.OnProcessTerminate()` callback, viene restituito un messaggio di errore. Per ulteriori informazioni sulla chiusura di un processo server, vedere [Rispondi a una notifica di chiusura del processo del server](#)

Esempio

```
Aws::GameLift::AwsLongOutcome TermTimeOutcome =  
    Aws::GameLift::Server::GetTerminationTime();
```

AcceptPlayerSession()

Notifica Amazon GameLift Servers che un giocatore con l'ID di sessione specificato si è connesso al processo del server e deve essere convalidato. Amazon GameLift Servers verifica che l'ID della sessione del giocatore sia valido. Dopo la convalida della sessione di gioco, Amazon GameLift Servers modifica lo stato dello slot del giocatore da RISERVATO ad ATTIVO.

Sintassi

```
GenericOutcome AcceptPlayerSession(String playerId)
```

Parameters

playerSessionId

ID univoco rilasciato Amazon GameLift Servers quando viene creata una nuova sessione giocatore.

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

Questo esempio gestisce una richiesta di connessione che include la convalida e il rifiuto di una sessione giocatore non valida. IDs

```
void ReceiveConnectingPlayerSessionID (Connection& connection, const std::string&
playerSessionId)
{
    Aws::GameLift::GenericOutcome connectOutcome =
    Aws::GameLift::Server::AcceptPlayerSession(playerSessionId);
    if(connectOutcome.IsSuccess())
    {
        connectionToSessionMap.emplace(connection, playerId);
        connection.Accept();
    }
    else
    {
        connection.Reject(connectOutcome.GetError().GetMessage());
    }
}
```

```
}
```

RemovePlayerSession()

Notifica Amazon GameLift Servers che un giocatore si è disconnesso dal processo del server. In risposta, Amazon GameLift Servers modifica lo slot del giocatore rendendolo disponibile.

Sintassi

```
GenericOutcome RemovePlayerSession(String playerId)
```

Parameters

playerSessionId

ID univoco rilasciato Amazon GameLift Servers quando viene creata una nuova sessione di gioco.

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

```
Aws::GameLift::GenericOutcome disconnectOutcome =  
    Aws::GameLift::Server::RemovePlayerSession(playerSessionId);
```

DescribePlayerSessions()

Recupera i dati della sessione del giocatore che includono impostazioni, metadati della sessione e dati del giocatore. Utilizzate questo metodo per ottenere informazioni su quanto segue:

- Una sessione per giocatore singolo
- Tutte le sessioni dei giocatori in una sessione di gioco
- Tutte le sessioni dei giocatori sono associate a un ID giocatore singolo

Sintassi

```
DescribePlayerSessionsOutcome DescribePlayerSessions(DescribePlayerSessionsRequest  
    describePlayerSessionsRequest)
```

Parameters

[DescribePlayerSessionsRequest](#)

Un [the section called “DescribePlayerSessionsRequest”](#) oggetto che descrive le sessioni dei giocatori da recuperare.

Valore restituito

Se l'esito è positivo, restituisce un oggetto [the section called “DescribePlayerSessionsOutcome”](#) contenente un set di oggetti di sessione giocatore corrispondente ai parametri della richiesta.

Esempio

Questo esempio richiede che tutte le sessioni dei giocatori siano connesse attivamente a una sessione di gioco specificata. Omettendo NextToken e impostando il valore Limite su 10, Amazon GameLift Servers restituisce i primi 10 record di sessione tra giocatori corrispondenti alla richiesta.

```
// Set request parameters
Aws::GameLift::Server::Model::DescribePlayerSessionsRequest request;
request.SetPlayerSessionStatusFilter(Aws::GameLift::Server::Model::PlayerSessionStatusMapper::G
request.SetLimit(10);
request.SetGameSessionId("the game session ID");    // can use GetGameSessionId()

// Call DescribePlayerSessions
Aws::GameLift::DescribePlayerSessionsOutcome playerSessionsOutcome =
    Aws::GameLift::Server::DescribePlayerSessions(request);
```

StartMatchBackfill()

Invia una richiesta per trovare nuovi giocatori per gli slot aperti in una sessione di gioco creata con FlexMatch. Per ulteriori informazioni, consulta la funzione di [FlexMatchriempimento](#).

Questa operazione è asincrona. Se vengono abbinati nuovi giocatori, Amazon GameLift Servers fornisce dati aggiornati sul matchmaker utilizzando la funzione di callback.

OnUpdateGameSession()

Un processo del server può avere un solo backfill degli abbinamenti attivo alla volta. Per inviare una nuova richiesta, chiama prima [StopMatchBackfill\(\)](#) per annullare la richiesta originale.

Sintassi

```
StartMatchBackfillOutcome StartMatchBackfill (StartMatchBackfillRequest
startBackfillRequest);
```

Parameters

[StartMatchBackfillRequest](#)

Un `StartMatchBackfillRequest` oggetto che comunica le seguenti informazioni:

- ID del ticket da assegnare alla richiesta di backfill. Queste informazioni sono facoltative; se non viene fornito alcun ID, ne Amazon GameLift Servers genererà uno.
- Matchmaker a cui inviare la richiesta. L'ARN di configurazione completo è obbligatorio. Questo valore si trova nei dati del matchmaker della sessione di gioco.
- L'ID della sessione di gioco da riempire.
- I dati di matchmaking disponibili per i giocatori attuali della sessione di gioco.

Valore restituito

Restituisce un [the section called “StartMatchBackfillOutcome”](#) oggetto con l'ID del ticket match backfill, oppure restituisce un errore con un messaggio di errore.

Esempio

```
// Build a backfill request
std::vector<Player> players;
Aws::GameLift::Server::Model::StartMatchBackfillRequest startBackfillRequest;
startBackfillRequest.SetTicketId("1111aaaa-22bb-33cc-44dd-5555eeee66ff"); // optional,
    autogenerated if not provided
startBackfillRequest.SetMatchmakingConfigurationArn("arn:aws:gamelift:us-
west-2:111122223333:matchmakingconfiguration/MyMatchmakerConfig"); //from the game
    session matchmaker data
startBackfillRequest.SetGameSessionArn("the game session ARN");           // can use
    GetGameSessionId()
startBackfillRequest.SetPlayers(players);                                 // from the
    game session matchmaker data

// Send backfill request
Aws::GameLift::StartMatchBackfillOutcome backfillOutcome =
    Aws::GameLift::Server::StartMatchBackfill(startBackfillRequest);
```

```
// Implement callback function for backfill
void Server::OnUpdateGameSession(Aws::GameLift::Server::Model::GameSession gameSession,
    Aws::GameLift::Server::Model::UpdateReason updateReason, std::string backfillTicketId)
{
    // handle status messages
    // perform game-specific tasks to prep for newly matched players
}
```

StopMatchBackfill()

Annulla una richiesta di match backfill attiva. Per ulteriori informazioni, consulta la funzione di [FlexMatchriempimento](#).

Sintassi

```
GenericOutcome StopMatchBackfill (StopMatchBackfillRequest stopBackfillRequest);
```

Parameters

[StopMatchBackfillRequest](#)

Un StopMatchBackfillRequest oggetto che identifica il ticket di matchmaking da annullare:

- L'ID del ticket assegnato alla richiesta di riempimento.
- Il matchmaker a cui è stata inviata la richiesta di riempimento.
- La sessione di gioco associata alla richiesta di riempimento.

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

```
// Set backfill stop request parameters

Aws::GameLift::Server::Model::StopMatchBackfillRequest stopBackfillRequest;
stopBackfillRequest.SetTicketId("1111aaaa-22bb-33cc-44dd-5555eeee66ff");
stopBackfillRequest.SetGameSessionArn("the game session ARN"); // can use
    GetGameSessionId()
stopBackfillRequest.SetMatchmakingConfigurationArn("arn:aws:gamelift:us-
west-2:111122223333:matchmakingconfiguration/MyMatchmakerConfig");
```

```
// from the game session matchmaker data

Aws::GameLift::GenericOutcome stopBackfillOutcome =
    Aws::GameLift::Server::StopMatchBackfill(stopBackfillRequest);
```

GetComputeCertificate()

Recupera il percorso del certificato TLS utilizzato per crittografare la connessione di rete tra la risorsa di elaborazione Amazon GameLift Servers Anywhere e Amazon GameLift Servers. Puoi utilizzare il percorso del certificato quando registri il tuo dispositivo di elaborazione in una flotta Anywhere. Amazon GameLift Servers Per ulteriori informazioni, consultare [RegisterCompute](#).

Sintassi

```
GetComputeCertificateOutcome Server::GetComputeCertificate()
```

Valore restituito

Restituisce una [the section called “GetComputeCertificateOutcome”](#).

Esempio

```
Aws::GameLift::GetComputeCertificateOutcome certificate =
    Aws::GameLift::Server::GetComputeCertificate();
```

GetFleetRoleCredentials()

Recupera le credenziali del ruolo IAM che autorizzano Amazon GameLift Servers a interagire con altri servizi AWS. Per ulteriori informazioni, consulta [Connect il server di gioco Amazon GameLift Servers ospitato ad altre AWS risorse](#).

Sintassi

```
GetFleetRoleCredentialsOutcome GetFleetRoleCredentials(GetFleetRoleCredentialsRequest request);
```

Parameters

[GetFleetRoleCredentialsRequest](#)

Valore restituito

Restituisce un oggetto [the section called “GetFleetRoleCredentialsOutcome”](#).

Esempio

```
// form the fleet credentials request
Aws::GameLift::Server::Model::GetFleetRoleCredentialsRequest
    getFleetRoleCredentialsRequest;
getFleetRoleCredentialsRequest.SetRoleArn("arn:aws:iam::123456789012:role/service-role/
exampleGameLiftAction");

Aws::GameLift::GetFleetRoleCredentialsOutcome credentials =
    Aws::GameLift::Server::GetFleetRoleCredentials(getFleetRoleCredentialsRequest);
```

Questo esempio mostra l'uso del RoleSessionName valore opzionale per assegnare un nome alla sessione di credenziali a fini di controllo. Se non si fornisce un nome per la sessione di ruolo, viene utilizzato il valore predefinito *[host-id] [fleet-id]-*».

```
// form the fleet credentials request
Aws::GameLift::Server::Model::GetFleetRoleCredentialsRequest
    getFleetRoleCredentialsRequest;
getFleetRoleCredentialsRequest.SetRoleArn("arn:aws:iam::123456789012:role/service-role/
exampleGameLiftAction");
getFleetRoleCredentialsRequest.SetRoleSessionName("MyFleetRoleSession");

Aws::GameLift::GetFleetRoleCredentialsOutcome credentials =
    Aws::GameLift::Server::GetFleetRoleCredentials(getFleetRoleCredentialsRequest);
```

Distruggi ()

Libera l'SDK del server di Amazon GameLift Servers gioco dalla memoria. È consigliabile chiamare questo metodo dopo ProcessEnding() e prima di terminare il processo. Se utilizzi una flotta Anywhere e non interrompi i processi del server dopo ogni sessione di gioco, chiama Destroy() e poi InitSDK() reinizializza prima di notificare Amazon GameLift Servers che il processo è pronto per ospitare una sessione di gioco. ProcessReady()

Sintassi

```
GenericOutcome Aws::GameLift::Server::Destroy();
```


Parameters

Non ci sono parametri.

Valore restituito

Restituisce un risultato generico costituito da successo o fallimento con un messaggio di errore.

Esempio

```
Aws::GameLift::GenericOutcome processEndingOutcome =  
    Aws::GameLift::Server::ProcessEnding();  
Aws::GameLift::Server::Destroy();  
  
// Exit the process with success or failure  
if (processEndingOutcome.IsSuccess()) {  
    exit(0);  
}  
else {  
    cout << "ProcessEnding() failed. Error: " <<  
    processEndingOutcome.GetError().GetErrorMessage();  
    exit(-1);  
}
```

Server C# SDK 5.x per -- Azioni Amazon GameLift Servers

Usa il riferimento all'SDK 5.x del server per integrare il tuo gioco multiplayer con cui ospitarlo.

Amazon GameLift Servers Per indicazioni sul processo di integrazione, consulta. [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#) Se stai usando il Amazon GameLift Servers plugin per Unity, vedi anche [Amazon GameLift Serversplugin per Unity \(server SDK 5.x\)](#).

Server C# SDK 5.x per Amazon GameLift Servers -- Tipi di dati

Usa il riferimento all'SDK 5.x del server Amazon GameLift Servers C# per integrare il tuo gioco multiplayer con cui ospitare. Amazon GameLift Servers Per indicazioni sul processo di integrazione, consulta. [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#) Se stai usando il Amazon GameLift Servers plugin per Unity, vedi anche [Amazon GameLift Serversplugin per Unity \(server SDK 5.x\)](#).

[Server C# SDK 5.x per -- Azioni Amazon GameLift Servers](#)

Tipi di dati

- [LogParameters](#)
- [MetricsParameters](#)
- [ProcessParameters](#)
- [UpdateGameSession](#)
- [GameSession](#)
- [ServerParameters](#)
- [StartMatchBackfillRequest](#)
- [Player](#)
- [DescribePlayerSessionsRequest](#)
- [StopMatchBackfillRequest](#)
- [GetFleetRoleCredentialsRequest](#)
- [AttributeValue](#)
- [AwsStringOutcome](#)
- [GenericOutcome](#)
- [MetricsManagerOutcome](#)
- [DescribePlayerSessionsOutcome](#)
- [DescribePlayerSessionsResult](#)
- [PlayerSession](#)
- [StartMatchBackfillOutcome](#)
- [StartMatchBackfillResult](#)
- [GetComputeCertificateOutcome](#)
- [GetComputeCertificateResult](#)
- [GetFleetRoleCredentialsOutcome](#)
- [GetFleetRoleCredentialsResult](#)
- [AwsDateTimeOutcome](#)
- [GameLiftError](#)
- [enumerazioni;](#)

LogParameters

Usa questo tipo di dati per identificare i file generati durante una sessione di gioco su cui desideri che il server di gioco venga caricato al Amazon GameLift Servers termine della sessione di gioco. Il server di gioco comunica `LogParameters` to Amazon GameLift Servers durante una [ProcessReady\(\)](#) chiamata.

Proprietà	Descrizione
<code>LogPaths</code>	L'elenco dei percorsi di directory dei file di registro del server di gioco che desideri Amazon GameLift Servers archiviare per accessi futuri. Il processo del server genera questi file durante ogni sessione di gioco. Definisci i percorsi e i nomi dei file nel tuo server di gioco e li memorizzi nella directory di build del gioco principale. I percorsi di registro devono essere assoluti. Ad esempio, se la build del gioco memorizza i log delle sessioni di gioco in un percorso simile <code>MyGame\sessionLogs\</code>, il percorso si troverebbe <code>c:\game\MyGame\sessionLogs</code> su un'istanza di Windows. Tipo: <code>List<String></code> Obbligatorio: no

MetricsParameters

Usa questo tipo di dati per configurare la raccolta delle metriche e la segnalazione degli arresti anomali per il server di gioco. Il server di gioco comunica con `MetricsParameters` durante una Amazon GameLift Servers [InitMetrics\(\)](#) chiamata.

Proprietà	Descrizione

StatsdHost	Il nome host o l'indirizzo IP del server StatSD per la raccolta delle metriche. Tipo: String Obbligatorio: no
StatsdPort	Il numero di porta del server StatSD per la raccolta delle metriche. Tipo: Integer Obbligatorio: no
CrashReporterHost	Il nome host o l'indirizzo IP del server Crash Reporter. Tipo: String Obbligatorio: no
CrashReporterPort	Il numero di porta del server Crash Reporter. Tipo: Integer Obbligatorio: no
FlushIntervalMs	L'intervallo in millisecondi per il trasferimento dei dati delle metriche al server. Tipo: Integer Obbligatorio: no
MaxPacketSize	La dimensione massima in byte per i pacchetti di metriche inviati al server. Tipo: Integer Obbligatorio: no

ProcessParameters

Questo tipo di dati contiene l'insieme di parametri a cui viene inviato una chiamata Amazon GameLift Servers. [ProcessReady\(\)](#)

Proprietà	Descrizione
LogParameters	L'oggetto con un elenco di percorsi di directory ai file di registro delle sessioni di gioco. Tipo: <code>Aws::GameLift::Server::</code> LogParameters Obbligatorio: sì
OnHealthCheck	Il nome della funzione di callback che Amazon GameLift Servers richiama per richiedere un rapporto sullo stato di salute del processo del server. Amazon GameLift Servers chiama questa funzione ogni 60 secondi. Dopo aver chiamato questa funzione Amazon GameLift Servers attende 60 secondi per una risposta, se non ne riceve nessuna, Amazon GameLift Servers registra il processo del server come non integro. Tipo: <code>void OnHealthCheckDelegate()</code> Obbligatorio: sì
OnProcessTerminate	Il nome della funzione di callback che Amazon GameLift Servers richiama per forzare l'arresto del processo server. Dopo aver richiamato questa funzione, Amazon GameLift Servers attende per cinque minuti l'arresto del processo del server e risponde con una chiamata ProcessEnding() prima di arrestare il processo del server.

	Tipo: void OnProcessTerminate Delegate() Obbligatorio: sì
OnStartGameSession	Il nome della funzione di callback che Amazon GameLift Servers richiama per attivare una nuova sessione di gioco. Amazon GameLift Servers richiama questa funzione in risposta alla richiesta del client. CreateGameSession La funzione di callback accetta un GameSession oggetto. Tipo: void OnStartGameSession Delegate(GameSession) Obbligatorio: sì
OnUpdateGameSession	Il nome della funzione di callback che Amazon GameLift Servers richiama per passare un oggetto di sessione di gioco aggiornato al processo del server. Amazon GameLift Servers richiama questa funzione quando è stata elaborata una richiesta di match backfill per fornire dati aggiornati sul matchmaker. Passa un GameSession oggetto, uno status update (updateReason) e l'ID del ticket match backfill. Tipo: void () OnUpdateGameSessionDelegate UpdateGameSession Obbligatorio: no

Porta

Il numero di porta su cui il processo server ascolta le connessioni di nuovi giocatori . Il valore deve rientrare nella gamma di porte configurate per qualsiasi parco istanze che distribuisce questa build del server di gioco. Questo numero di porta è incluso nella sessione di gioco e negli oggetti della sessione del giocatore utilizzati dalle sessioni di gioco per la connessione a un processo server.

Tipo: Integer

Obbligatorio: sì

UpdateGameSession

Le informazioni aggiornate relative a un oggetto della sessione di gioco includono il motivo per cui la sessione di gioco è stata aggiornata. Se l'aggiornamento è correlato a un'azione di ripristino di una partita, questo tipo di dati include l'ID del ticket di backfill.

Properties	Descrizione
GameSession	Oggetto GameSession. L'GameSession oggetto contiene proprietà che descrivono una sessione di gioco. Tipo: GameSession GameSession() Obbligatorio: sì
UpdateReason	Il motivo per cui la sessione di gioco viene aggiornata. Tipo: UpdateReason UpdateReason() Obbligatorio: sì
BackfillTicketId	L'ID del ticket di backfill che tenta di aggiornare la sessione di gioco.

Properties	Descrizione
	Tipo: String
	Obbligatorio: sì

GameSession

Dettagli di una sessione di gioco.

Properties	Descrizione
GameSessionId	Un identificatore univoco per la sessione di gioco. L'ARN di una sessione di gioco ha il seguente formato: <code>arn:aws:gamelift:<region>::gamesession/<fleet ID>/<custom ID string or idempotency token></code> Tipo: String Obbligatorio: no
Name	Un'etichetta descrittiva della sessione di gioco. Tipo: String Obbligatorio: no
FleetId	Un identificatore univoco per la flotta su cui è in esecuzione la sessione di gioco. Tipo: String Obbligatorio: no
MaximumPlayerSessionCount	Il numero massimo di connessioni dei giocatori alla sessione di gioco. Tipo: Integer

Properties	Descrizione
	Obbligatorio: no
Porta	Il numero di porta per la sessione di gioco. Per connettersi a un server di Amazon GameLift Servers gioco, un'app necessita sia dell'indirizzo IP che del numero di porta. Tipo: Integer Obbligatorio: no
IpAddress	L'indirizzo IP della sessione di gioco. Per connettersi a un server di Amazon GameLift Servers gioco, un'app necessita sia dell'indirizzo IP che del numero di porta. Tipo: String Obbligatorio: no
GameSessionData	Un set di proprietà personalizzate della sessione di gioco, formattate come valore di stringa singola. Tipo: String Obbligatorio: no

Properties	Descrizione
MatchmakerData	Le informazioni sul processo di matchmaking utilizzato per creare la sessione di gioco, in sintassi JSON, sono formattate come stringa. Oltre alla configurazione di matchmaking utilizzata, contiene dati su tutti i giocatori assegnati alla partita, compresi gli attributi dei giocatori e le assegnazioni delle squadre. Tipo: <code>String</code> Obbligatorio: no
GameProperties	Un insieme di proprietà personalizzate per una sessione di gioco, formattate come coppie chiave:valore. Queste proprietà vengono passate con una richiesta di avvio di una nuova sessione di gioco. Tipo: <code>Dictionary<string, string></code> Obbligatorio: no

Properties	Descrizione
DnsName	L'identificatore DNS assegnato all'istanza che esegue la sessione di gioco. I valori hanno il seguente formato: • Flotte abilitate per TLS: <code><unique identifier>.<region identifier>.amazongamelift.com</code> • Non-TLS-enabled flotte: <code>ec2-<unique identifier>.compute.amazonaws.com</code> Quando ti connetti a una sessione di gioco in esecuzione su una flotta abilitata per TLS, devi usare il nome DNS, non l'indirizzo IP. Tipo: String Obbligatorio: no

ServerParameters

Informazioni utilizzate per mantenere la connessione tra un server Amazon GameLift Servers Anywhere e il servizio. Amazon GameLift Servers Queste informazioni vengono utilizzate quando si avviano nuovi processi server con [InitSDK\(\)](#). Per i server ospitati su istanze EC2 Amazon GameLift Servers gestite, utilizza un oggetto vuoto.

Properties	Descrizione
WebSocketUrl	Il <code>GameLiftServerSdkEndpoint</code> restituito quando fai parte <code>RegisterCompute</code> di Anywhere. Amazon GameLift Servers Tipo: String Obbligatorio: sì

Properties	Descrizione
ProcessId	Un identificatore univoco registrato nel processo del server che ospita il gioco. Tipo: <code>String</code> Obbligatorio: sì
HostId	Un identificatore univoco per l'host con i processi del server che ospitano il gioco. L'HostID viene <code>ComputeName</code> utilizzato quando hai registrato il tuo computer. Per ulteriori informazioni, consulta, RegisterCompute Tipo: <code>String</code> Obbligatorio: sì
FleetId	L'ID del parco veicoli su cui è registrato il computer. Per ulteriori informazioni, consultare RegisterCompute. Tipo: <code>String</code> Obbligatorio: sì
AuthToken	Il token di autenticazione generato da Amazon GameLift Servers questo codice autentica il server. Amazon GameLift Servers Per ulteriori informazioni, consultare GetComputeAuthToken. Tipo: <code>String</code> Obbligatorio: sì

StartMatchBackfillRequest

Informazioni utilizzate per creare una richiesta di matchmaking backfill. Il server di gioco comunica queste informazioni durante una chiamata Amazon GameLift Servers. [StartMatchBackfill\(\)](#)

Properties	Descrizione
GameSessionArn	L'identificatore univoco della sessione di gioco. L'operazione API GetGameSessionId restituisce l'identificatore in formato ARN. Tipo: String Obbligatorio: sì
MatchmakingConfigurationArn	L'identificatore univoco, sotto forma di ARN, che il matchmaker deve utilizzare per questa richiesta. L'ARN del matchmaker per la sessione di gioco originale si trova nell'oggetto della sessione di gioco nella proprietà matchmaker data. Scopri di più sui dati dei matchmaker in Work with matchmaker data. Tipo: String Obbligatorio: sì
Players	Un insieme di dati che rappresenta tutti i giocatori attualmente impegnati nella sessione di gioco. Il matchmaker utilizza queste informazioni per cercare nuovi giocatori che rappresentano un buon abbinamento per i giocatori attuali. Tipo: List<Player> Obbligatorio: sì
TicketId	L'identificatore univoco di un ticket di richiesta di matchmaking o match backfill. Se non

Properties	Descrizione
	fornisci un valore, Amazon GameLift Servers ne genera uno. Utilizzare questo identificatore per monitorare lo stato del ticket di backfill degli abbinamenti o annullare la richiesta, se necessario. Tipo: <code>String</code> Obbligatorio: no

Player

Rappresenta un giocatore in matchmaking. Quando inizia una richiesta di matchmaking, un giocatore ha un ID giocatore, attributi e possibilmente dati sulla latenza. Amazon GameLift Servers aggiunge le informazioni sulla squadra dopo una partita.

Properties	Descrizione
LatencyInSM	Un insieme di valori espressi in millisecondi, che indica la quantità di latenza che un giocatore sperimenta quando è connesso a una posizione. Se viene utilizzata questa proprietà, il giocatore viene abbinato solo per le posizioni elencate. Se un matchmaker ha una regola che valuta la latenza dei giocatori, i giocatori devono segnalare la latenza per essere abbinati. Tipo: <code>Dictionary<string, int></code> Obbligatorio: no
PlayerAttributes	Una raccolta di coppie chiave:valore che contengono informazioni sui giocatori da utilizzare nel matchmaking. Le chiavi degli attributi del giocatore devono corrispondere a

Properties	Descrizione
	quelle PlayerAttributes utilizzate in un set di regole di matchmaking. Per ulteriori informazioni sugli attributi dei giocatori, vedi AttributeValue. Tipo: Dictionary<string, Attribute Value Obbligatorio: no
PlayerId	Un identificatore univoco per un giocatore. Tipo: String Obbligatorio: no
Team	Il nome della squadra a cui è assegnato il giocatore in una partita. Definisci il nome della squadra nel set di regole del matchmaking. Tipo: String Obbligatorio: no

DescribePlayerSessionsRequest

Questo tipo di dati viene utilizzato per specificare quale sessione del giocatore recuperare. Può essere utilizzato in diversi modi: (1) fornire PlayerSessionId a per richiedere una sessione di gioco specifica; (2) fornire GameSessionId a richiedere tutte le sessioni di gioco nella sessione di gioco specificata; o (3) fornire PlayerId a per richiedere tutte le sessioni di gioco per il giocatore specificato. Per le raccolte di grandi dimensioni delle sessioni giocatore, utilizzare i parametri di paginazione per recuperare i risultati in pagine sequenziali.

Properties	Descrizione
GameSessionId	L'identificatore unico della sessione di gioco. Utilizzare questo parametro per richiedere

Properties	Descrizione
	tutte le sessioni giocatore per la sessione di gioco specificata. Il formato dell'ID della sessione di gioco è il seguente: <code>arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string></code> . Il valore di <code><ID string></code> corrisponde a una stringa ID personalizzata o (se ne è stata specificata una al momento della creazione della sessione di gioco) a una stringa generata. Tipo: <code>String</code> Obbligatorio: no
PlayerSessionId	L'identificatore univoco per la sessione di un giocatore. Tipo: <code>String</code> Obbligatorio: no
PlayerId	L'identificatore univoco di un giocatore. Per informazioni, consulta Genera giocatore IDs. Tipo: <code>String</code> Obbligatorio: no

Properties	Descrizione
PlayerSessionStatusFilter	Lo stato della sessione del giocatore su cui filtrare i risultati. Tra gli stati sessione giocatore possibili sono inclusi i seguenti: • RISERVATO — La richiesta di sessione del giocatore è stata ricevuta, ma il giocatore non si è ancora connesso al server. La procedura and/or è stata convalidata. • ACTIVE (ATTIVO) - Il giocatore è stato convalidato dal processo del server ed è attualmente collegato. • COMPLETED (COMPLETATO) - La connessione del giocatore è stata interrotta. • TIMEDOUT — È stata ricevuta una richiesta di sessione di gioco, ma il giocatore che non si and/or è connesso non è stata convalidata entro il limite di timeout (60 secondi). Tipo: <code>String</code> Obbligatorio: no
NextToken	Il token che indica l'inizio della pagina successiva di risultati. Per specificare l'inizio del set di risultati, non fornire un valore. Se fornisci un ID di sessione del giocatore, questo parametro viene ignorato. Tipo: <code>String</code> Obbligatorio: no

Properties	Descrizione
Limite	Numero massimo di risultati da restituire. Se fornisci un ID di sessione del giocatore, questo parametro viene ignorato. Tipo: <code>int</code> Obbligatorio: no

StopMatchBackfillRequest

Informazioni utilizzate per annullare una richiesta di matchmaking backfill. Il server di gioco comunica queste informazioni al Amazon GameLift Servers servizio durante una chiamata. [StopMatchBackfill\(\)](#)

Properties	Descrizione
GameSessionArn	L'identificatore univoco della sessione di gioco della richiesta annullata. Tipo: <code>string</code> Obbligatorio: sì
MatchmakingConfigurationArn	L'identificatore univoco del matchmaker a cui è stata inviata questa richiesta. Tipo: <code>string</code> Obbligatorio: sì
TicketId	L'identificatore univoco del ticket di richiesta di riempimento da annullare. Tipo: <code>string</code> Obbligatorio: sì

GetFleetRoleCredentialsRequest

Questo tipo di dati offre al server di gioco un accesso limitato alle altre risorse. AWS Per ulteriori informazioni, consultare [Configurare un ruolo di servizio IAM per Amazon GameLift Servers](#).

Properties	Descrizione
RoleArn	L'Amazon Resource Name (ARN) del ruolo di servizio che estende l'accesso limitato alle tue AWS risorse. Tipo: <code>string</code> Obbligatorio: sì
RoleSessionName	Il nome della sessione che descrive l'uso delle credenziali del ruolo. Tipo: <code>string</code> Obbligatorio: no

AttributeValue

Usa questi valori nelle coppie chiave-valore degli attributi. [Player](#) Questo oggetto consente di specificare il valore di un attributo utilizzando uno qualsiasi dei tipi di dati validi: stringa, numero, array di stringhe o mappa di dati. Ogni `AttributeValue` oggetto può utilizzare solo una delle proprietà disponibili.

Properties	Description
AttrType	Specifica il tipo di valore dell'attributo. Tipo: un valore <code>AttrType</code> enum. Obbligatorio: no
S	Rappresenta un valore di attributo di tipo stringa. Tipo: <code>string</code>

Properties	Description
	Obbligatorio: sì
N	Rappresenta un valore di attributo numerico. Tipo: <code>double</code> Obbligatorio: sì
SL	Rappresenta una matrice di valori di attributi di tipo stringa. Tipo: <code>string[]</code> Obbligatorio: sì
SDM	Rappresenta un dizionario di chiavi di stringa e valori doppi. Tipo: <code>Dictionary<string, double></code> Obbligatorio: sì

AwsStringOutcome

Questo tipo di dati risulta da un'azione e produce un oggetto con le seguenti proprietà:

Properties	Description
Risultato	Il risultato dell'azione. Tipo: <code>string</code> Obbligatorio: no
Completato	Se l'azione ha avuto successo o meno. Tipo: <code>bool</code> Obbligatorio: sì
Errore	L'errore che si è verificato se l'azione non ha avuto successo.

Properties	Description
	Tipo: the section called “GameLiftError”
	Obbligatorio: no

GenericOutcome

Questo tipo di dati risulta da un'azione e produce un oggetto con le seguenti proprietà:

Properties	Description
Completato	Se l'azione ha avuto successo o meno. Tipo: bool Obbligatorio: sì
Errore	L'errore che si è verificato se l'azione non ha avuto successo. Tipo: the section called “GameLiftError” Obbligatorio: no

MetricsManagerOutcome

Rappresenta il risultato di una [InitMetrics\(\)](#) chiamata. Contiene un' MetricsManager istanza relativa all'esito positivo o informazioni sull'errore in caso di errore.

Risultato	L' MetricsManager istanza per la raccolta e la segnalazione delle metriche. Tipo: MetricsManager Obbligatorio: no
Completato	Indipendentemente dal fatto che l'azione abbia avuto successo o meno. Tipo: bool

	Obbligatorio: sì
Errore	L'errore che si è verificato se l'azione non ha avuto successo. Tipo: the section called “GameLiftError” Obbligatorio: no

DescribePlayerSessionsOutcome

Questo tipo di dati risulta da un'azione e produce un oggetto con le seguenti proprietà:

Properties	Description
Risultato	Il risultato dell'azione. Tipo: the section called “DescribePlayerSessionsResult” Obbligatorio: no
Completato	Se l'azione ha avuto successo o meno. Tipo: bool Obbligatorio: sì
Errore	L'errore che si è verificato se l'azione non ha avuto successo. Tipo: the section called “GameLiftError” Obbligatorio: no

DescribePlayerSessionsResult

Properties	Description
NextToken	Il token che indica l'inizio della pagina successiva dei risultati. Per specificare l'inizio del set di risultati, non fornire un valore.

Properties	Description
	Se fornisci un ID di sessione del giocatore, questo parametro viene ignorato. Tipo: <code>string</code> Obbligatorio: sì
PlayerSessions	Una raccolta di oggetti contenente proprietà per ogni sessione di giocatore che corrisponde alla richiesta. Tipo: <code>IList<the section called "PlayerSession"></code> Campo obbligatorio:
Completato	Se l'azione ha avuto successo o meno. Tipo: <code>bool</code> Obbligatorio: sì
Errore	L'errore che si è verificato se l'azione non ha avuto successo. Tipo: the section called "GameLiftError" Obbligatorio: no

PlayerSession

Properties	Description
CreationTime	Tipo: <code>long</code> Obbligatorio: sì
FleetId	Tipo: <code>string</code> Obbligatorio: sì
GameSessionId	Tipo: <code>string</code>

Properties	Description
	Obbligatorio: sì
IpAddress	Tipo: <code>string</code> Obbligatorio: sì
PlayerData	Tipo: <code>string</code> Obbligatorio: sì
PlayerId	Tipo: <code>string</code> Obbligatorio: sì
PlayerSessionId	Tipo: <code>string</code> Obbligatorio: sì
Porta	Tipo: <code>int</code> Obbligatorio: sì
Status	Tipo: Un <code>PlayerSessionStatus</code> enum . Obbligatorio: sì
TerminationTime	Tipo: <code>long</code> Obbligatorio: sì
DnsName	Tipo: <code>string</code> Obbligatorio: sì

StartMatchBackfillOutcome

Questo tipo di dati risulta da un'azione e produce un oggetto con le seguenti proprietà:

Properties	Description
Risultato	Il risultato dell'azione. Tipo: the section called “StartMatchBackfillResult” Obbligatorio: no
Completato	Se l'azione ha avuto successo o meno. Tipo: bool Obbligatorio: sì
Errore	L'errore che si è verificato se l'azione non ha avuto successo. Tipo: the section called “GameLiftError” Obbligatorio: no

StartMatchBackfillResult

Properties	Description
TicketId	Tipo: string Obbligatorio: sì

GetComputeCertificateOutcome

Questo tipo di dati risulta da un'azione e produce un oggetto con le seguenti proprietà:

Properties	Description
Risultato	Il risultato dell'azione. Tipo: the section called “GetComputeCertificateResult” Obbligatorio: no

Properties	Description
Completato	Se l'azione ha avuto successo o meno. Tipo: bool Obbligatorio: sì
Errore	L'errore che si è verificato se l'azione non ha avuto successo. Tipo: the section called “GameLiftError” Obbligatorio: no

GetComputeCertificateResult

Il percorso del certificato TLS sul computer e il nome host del computer.

Properties	Description
CertificatePath	Tipo: string Obbligatorio: sì
ComputeName	Tipo: string Obbligatorio: sì

GetFleetRoleCredentialsOutcome

Questo tipo di dati risulta da un'azione e produce un oggetto con le seguenti proprietà:

Properties	Description
Risultato	Il risultato dell'azione. Tipo: the section called “GetFleetRoleCredentialsResult” Obbligatorio: no

Properties	Description
Completato	Se l'azione ha avuto successo o meno. Tipo: bool Obbligatorio: sì
Errore	L'errore che si è verificato se l'azione non ha avuto successo. Tipo: the section called “GameLiftError” Obbligatorio: no

GetFleetRoleCredentialsResult

Properties	Description
AccessKeyId	L'ID della chiave di accesso per autenticare e fornire l'accesso alle tue risorse. AWS Tipo: string Obbligatorio: no
AssumedRoleId	L'ID dell'utente a cui appartiene il ruolo di servizio. Tipo: string Obbligatorio: no
AssumedRoleUserArn	L'Amazon Resource Name (ARN) dell'utente a cui appartiene il ruolo di servizio. Tipo: string Obbligatorio: no
Scadenza	Il periodo di tempo che manca alla scadenza delle credenziali di sessione.

Properties	Description
	Tipo: <code>DateTime</code> Obbligatorio: no
<code>SecretAccessKey</code>	L'ID della chiave di accesso segreta per l'autenticazione. Tipo: <code>string</code> Obbligatorio: no
<code>SessionToken</code>	Un token per identificare la sessione attiva corrente che interagisce con AWS le tue risorse. Tipo: <code>string</code> Obbligatorio: no
<code>Completato</code>	Se l'azione ha avuto successo o meno. Tipo: <code>bool</code> Obbligatorio: sì
<code>Errore</code>	L'errore che si è verificato se l'azione non ha avuto successo. Tipo: the section called “GameLiftError” Obbligatorio: no

AwsDateTimeOutcome

Questo tipo di dati risulta da un'azione e produce un oggetto con le seguenti proprietà:

Properties	Description
<code>Risultato</code>	Il risultato dell'azione. Tipo: <code>DateTime</code>

Properties	Description
	Obbligatorio: no
Completato	Se l'azione ha avuto successo o meno. Tipo: <code>bool</code> Obbligatorio: sì
Errore	L'errore che si è verificato se l'azione non ha avuto successo. Tipo: the section called “GameLiftError” Obbligatorio: no

GameLiftError

Properties	Description
ErrorType	Il tipo di errore. Tipo: Un enum. <code>GameLiftErrorType</code> Obbligatorio: no
ErrorMessage	Il nome dell'errore. Tipo: <code>string</code> Obbligatorio: no
ErrorMessage	Messaggio di errore. Tipo: <code>string</code> Obbligatorio: no

enumerazioni;

Le enumerazioni definite per l'SDK del server per Amazon GameLift Servers (C#) sono definite come segue:

AttrType

- NONE
- STRINGA
- DOPPIA
- ELENCO_STRINGHE
- STRING_DOUBLE_MAP

GameLiftErrorType

Valore di stringa che indica il tipo di errore. I valori validi includono:

- SERVICE_CALL_FAILED — Una chiamata a un servizio non è riuscita. AWS
- LOCAL_CONNECTION_FAILED — La connessione locale a non è riuscita. Amazon GameLift Servers
- NETWORK_NOT_INITIALIZED — La rete non è stata inizializzata.
- GAMESESSION_ID_NOT_SET — L'ID della sessione di gioco non è stato impostato.
- BAD_REQUEST_EXCEPTION
- ECCEZIONE_DI_SERVIZIO INTERNA
- ALREADY_INITIALIZED — Il Amazon GameLift Servers server o il client è già stato inizializzato con Initialize ().
- FLEET_MISMATCH — La flotta bersaglio non corrisponde alla flotta di una GameSession o PlayerSession.
- GAMELIFT_CLIENT_NOT_INITIALIZED — Il client non è stato inizializzato. Amazon GameLift Servers
- GAMELIFT_SERVER_NOT_INITIALIZED — Amazon GameLift Servers Il server non è stato inizializzato.
- GAME_SESSION_ENDED_FAILED: l'SDK del server non è riuscito a contattare il servizio per segnalare la fine della sessione di gioco. Amazon GameLift Servers
- GAME_SESSION_NOT_READY — La sessione di gioco sul server non è stata attivata. Amazon GameLift Servers

- `GAME_SESSION_READY_FAILED`: l'SDK del server per cui non è stato possibile contattare il servizio per segnalare che la sessione di gioco è pronta. Amazon GameLift Servers
- `INITIALIZATION_MISMATCH` — È stato chiamato un metodo client dopo `Server: :Initialize ()` o viceversa.
- `NOT_INITIALIZED` — Il Amazon GameLift Servers server o il client non sono stati inizializzati con `Initialize ()`.
- `NO_TARGET_ALIASID_SET` — Non è stato impostato un `aliasID` di destinazione.
- `NO_TARGET_FLEET_SET` — Non è stata impostata una flotta di obiettivi.
- `PROCESS_ENDING_FAILED`: l'SDK del server non è riuscito a contattare il servizio per segnalare la fine del processo. Amazon GameLift Servers
- `PROCESS_NOT_ACTIVE` — Il processo del server non è ancora attivo, non è associato a un processo e non può accettarlo o elaborarlo. `GameSession PlayerSessions`
- `PROCESS_NOT_READY` — Il processo del server non è ancora pronto per essere attivato.
- `PROCESS_READY_FAILED`: l'SDK del server non è riuscito a contattare il servizio per Amazon GameLift Servers segnalare che il processo è pronto.
- `SDK_VERSION_DETECTION_FAILED` — Rilevamento della versione SDK non riuscito.
- `STX_CALL_FAILED` — Una chiamata al componente di backend del server non è riuscita. `XStx`
- `STX_INITIALIZATION_FAILED` — L'inizializzazione del componente di backend del server non è riuscita. `XStx`
- `UNEXPECTED_PLAYER_SESSION` — Il server ha rilevato una sessione di giocatore non registrata.
- `WEBSOCKET_CONNECT_FAILURE`
- `WEBSOCKET_CONNECT_FAILURE_FORBIDDEN`
- `URL NON VALIDO DI WEBSOCKET_CONNECT_FAILURE_INVALID_`
- `WEBSOCKET_CONNECT_FAILURE_TIMEOUT`
- `WEBSOCKET_RETRIABLE_SEND_MESSAGE_FAILURE` — Errore recuperabile nell'invio di un messaggio al servizio. `GameLift WebSocket`
- `WEBSOCKET_SEND_MESSAGE_FAILURE` — Errore nell'invio di un messaggio al GameLift servizio. `WebSocket`
- `MATCH_BACKFILL_REQUEST_VALIDATION` — La convalida della richiesta non è riuscita.
- `PLAYER_SESSION_REQUEST_VALIDATION` — La convalida della richiesta non è riuscita.

PlayerSessionCreationPolicy

Valore della stringa che indica se la sessione di gioco accetta nuovi giocatori. I valori validi includono:

- `ACCEPT_ALL`: accetta tutte le nuove sessioni giocatore.
- `DENY_ALL`: rifiuta tutte le nuove sessioni giocatore.
- `NOT_SET` — La sessione di gioco non è impostata per accettare o rifiutare sessioni di nuovi giocatori.

PlayerSessionStatus

- `ACTIVE`
- `COMPLETED`
- `NOT_SET`
- `RISERVATO`
- `TIMEOUT`

[Server C# SDK 5.x per Amazon GameLift Servers -- Tipi di dati](#)

Argomenti

- [GetSdkVersion\(\)](#)
- [InitMetrics\(\)](#)
- [InitSDK\(\)](#)
- [InitSDK\(\)](#)
- [ProcessReady\(\)](#)
- [ProcessEnding\(\)](#)
- [ActivateGameSession\(\)](#)
- [UpdatePlayerSessionCreationPolicy\(\)](#)
- [GetGameSessionId\(\)](#)
- [GetTerminationTime\(\)](#)
- [AcceptPlayerSession\(\)](#)
- [RemovePlayerSession\(\)](#)
- [DescribePlayerSessions\(\)](#)
- [StartMatchBackfill\(\)](#)

- [StopMatchBackfill\(\)](#)
- [GetComputeCertificate\(\)](#)
- [GetFleetRoleCredentials\(\)](#)
- [Distruggi \(\)](#)

GetSdkVersion()

Restituisce il numero di versione corrente dell'SDK integrato nel processo del server.

Sintassi

```
AwsStringOutcome GetSdkVersion();
```

Valore restituito

Se l'esito è positivo, restituisce la versione corrente dell'SDK come oggetto [the section called "AwsStringOutcome"](#). La stringa restituita include il numero di versione (esempio `5.0.0`). Se l'esito è negativo, verrà restituito un messaggio di errore.

Esempio

```
var getSdkVersionOutcome = GameLiftServerAPI.GetSdkVersion();
```

InitMetrics()

Inizializza il sistema di metriche per la raccolta e la segnalazione dei dati sulle prestazioni del server. Per ottenere i migliori risultati, chiamate prima questo metodo per abilitare la raccolta delle metriche [InitSDK\(\)](#) durante l'inizializzazione dell'SDK.

Sintassi

```
MetricsManagerOutcome InitMetrics();  
MetricsManagerOutcome InitMetrics(MetricsParameters metricsParameters);
```

Parameters

[MetricsParameters](#) (facoltativo)

Parametri di configurazione per la raccolta delle metriche. Se non viene fornito, utilizza valori predefiniti che possono essere sostituiti dalle variabili di ambiente.

Valore restituito

In caso di successo, restituisce un [the section called "MetricsManagerOutcome"](#) oggetto contenente l' `MetricsManager` istanza. Se l'esito è negativo, verrà restituito un messaggio di errore.

Esempio

```
// Initialize with default parameters (uses environment variables if available)
var outcome = GameLiftServerAPI.InitMetrics();
if (outcome.Success) {
    var metricsManager = outcome.Result;
} else {
    Console.WriteLine("Failed to initialize metrics: " + outcome.Error.ErrorMessage);
}

// Initialize with custom parameters
var metricsParams = new MetricsParameters("localhost", 8125, "crash-host", 9999, 1000,
    1024);
var customOutcome = GameLiftServerAPI.InitMetrics(metricsParams);
if (customOutcome.Success) {
    var metricsManager = customOutcome.Result;
}
```

InitSDK()

Inizializza l'Amazon GameLift Servers SDK per una flotta EC2 gestita. Richiama questo metodo all'avvio, prima che si verifichi qualsiasi altra inizializzazione correlata a. Amazon GameLift Servers. Questo metodo legge i parametri del server dall'ambiente host per configurare la comunicazione tra il server e il Amazon GameLift Servers servizio. Utilizza un token di idempotenza, quindi puoi riprovare a eseguire questa chiamata in tutta sicurezza quando fallisce.

Sintassi

```
GenericOutcome InitSDK();
```

Valore restituito

In caso di successo, restituisce un `InitSdkOutcome` oggetto per indicare che il processo del server è pronto per la chiamata. [ProcessReady\(\)](#)

Esempio

```
//Call InitSDK to establish a local connection with the GameLift agent to enable
further communication.
GenericOutcome initSDKOutcome = GameLiftServerAPI.InitSDK();
```

InitSDK()

Inizializza l'Amazon GameLift ServersSDK per una flotta Anywhere. Richiama questo metodo all'avvio, prima che si verifichi qualsiasi altra inizializzazione correlata a. Amazon GameLift Servers. Questo metodo richiede parametri espliciti del server per configurare la comunicazione tra il server e il Amazon GameLift Servers servizio. Utilizza un token di idempotenza, quindi puoi riprovare a ripetere questa chiamata in tutta sicurezza quando fallisce.

Sintassi

```
GenericOutcome InitSDK(ServerParameters serverParameters);
```

Parameters

[ServerParameters](#)

Per inizializzare un server di gioco su una flotta Amazon GameLift Servers Anywhere, costruisci un `ServerParameters` oggetto con le seguenti informazioni:

- L'URL WebSocket utilizzato per connettersi al server di gioco.
- L'ID del processo utilizzato per ospitare il server di gioco.
- L'ID del computer che ospita i processi del server di gioco.
- L'ID della Amazon GameLift Servers flotta contenente il tuo computer Amazon GameLift Servers Anywhere.
- Il token di autorizzazione generato dall'Amazon GameLift Serversoperazione.

Valore restituito

In caso di successo, restituisce un `InitSdkOutcome` oggetto per indicare che il processo del server è pronto per la chiamata [ProcessReady\(\)](#).

 Note

Se le chiamate a non `InitSDK()` riescono per le build di gioco distribuite sulle flotte Anywhere, controlla il `ServerSdkVersion` parametro utilizzato durante la creazione della risorsa di compilazione. È necessario impostare esplicitamente questo valore sulla versione SDK del server in uso. Il valore predefinito per questo parametro è 4.x, che non è compatibile. Per risolvere questo problema, crea una nuova build e distribuiscila in una nuova flotta.

Esempio

```
//Define the server parameters
string websocketUrl = "wss://us-west-1.api.amazongamelift.com";
string processId = "PID1234";
string fleetId = "aarn:aws:gamelift:us-west-1:111122223333:fleet/
fleet-9999ffff-88ee-77dd-66cc-5555bbbb44aa";
string hostId = "HardwareAnywhere";
string authToken = "1111aaaa-22bb-33cc-44dd-5555eeee66ff";
ServerParameters serverParameters =
    new ServerParameters(websocketUrl, processId, hostId, fleetId, authToken);

//Call InitSDK to establish a local connection with the GameLift agent to enable
    further communication.
GenericOutcome initSDKOutcome = GameLiftServerAPI.InitSDK(serverParameters);
```

ProcessReady()

Notifica Amazon GameLift Servers che il processo del server è pronto per ospitare sessioni di gioco. Chiama questo metodo dopo averlo [InitSDK\(\)](#) invocato. Questo metodo deve essere chiamato solo una volta per processo.

Sintassi

```
GenericOutcome ProcessReady(ProcessParameters processParameters)
```

Parameters

[ProcessParameters](#)

Un `ProcessParameters` oggetto contiene informazioni sul processo del server.

Valore restituito

Restituisce un risultato generico costituito da successo o fallimento con un messaggio di errore.

Esempio

Questo esempio illustra sia l'implementazione del metodo che quella delle funzioni di delega.

```
// Set parameters and call ProcessReady
ProcessParameters processParams = new ProcessParameters(
    this.OnStartGameSession,
    this.OnProcessTerminate,
    this.OnHealthCheck,
    this.OnUpdateGameSession,
    port,
    new LogParameters(new List<string>()
// Examples of log and error files written by the game server
{
    "C:\\game\\logs",
    "C:\\game\\error"
}))
);
GenericOutcome processReadyOutcome = GameLiftServerAPI.ProcessReady(processParams);
```

ProcessEnding()

Notifica Amazon GameLift Servers che il processo del server sta terminando. Chiama questo metodo dopo tutte le altre attività di pulizia (inclusa la chiusura della sessione di gioco attiva) e prima di terminare il processo. A seconda del risultato di `ProcessEnding()`, il processo termina con successo (0) o errore (-1) e genera un evento relativo alla flotta. Se il processo termina con un errore, l'evento fleet generato è. `SERVER_PROCESS_TERMINATED_UNHEALTHY`

Sintassi

```
GenericOutcome ProcessEnding()
```

Valore restituito

Restituisce un risultato generico costituito da successo o fallimento con un messaggio di errore.

Esempio

Questo esempio chiama `ProcessEnding()` e `Destroy()` prima di terminare il processo del server con un codice di uscita riuscito o di errore.

```
GenericOutcome processEndingOutcome = GameLiftServerAPI.ProcessEnding();
GameLiftServerAPI.Destroy();

if (processEndingOutcome.Success)
{
    Environment.Exit(0);
}
else
{
    Console.WriteLine("ProcessEnding() failed. Error: " +
processEndingOutcome.Error.ToString());
    Environment.Exit(-1);
}
```

ActivateGameSession()

Notifica Amazon GameLift Servers che il processo del server ha attivato una sessione di gioco ed è ora pronto a ricevere le connessioni dei giocatori. Questa azione dovrebbe essere richiamata come parte della funzione di `onStartGameSession()` callback, dopo l'inizializzazione di tutta la sessione di gioco.

Sintassi

```
GenericOutcome ActivateGameSession()
```

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

Questo esempio illustra la chiamata a `ActivateGameSession()` nell'ambito della funzione delegata `onStartGameSession()`.

```
void OnStartGameSession(GameSession gameSession)
{
    // game-specific tasks when starting a new game session, such as loading map
```

```
// When ready to receive players
GenericOutcome activateGameSessionOutcome = GameLiftServerAPI.ActivateGameSession();
}
```

UpdatePlayerSessionCreationPolicy()

Aggiorna la capacità della sessione di gioco corrente di accettare nuove sessioni giocatore. Una sessione di gioco può essere configurata per accettare o rifiutare tutte le nuove sessioni giocatore.

Sintassi

```
GenericOutcome UpdatePlayerSessionCreationPolicy(PlayerSessionCreationPolicy
playerSessionPolicy)
```

Parameters

playerSessionPolicy

Valore di stringa che indica se la sessione di gioco accetta nuovi giocatori.

I valori validi includono:

- **ACCEPT_ALL**: accetta tutte le nuove sessioni giocatore.
- **DENY_ALL**: rifiuta tutte le nuove sessioni giocatore.

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

Questo esempio definisce la policy di partecipazione alla sessione di gioco corrente per accettare tutti i giocatori.

```
GenericOutcome updatePlayerSessionPolicyOutcome =
    GameLiftServerAPI.UpdatePlayerSessionCreationPolicy(PlayerSessionCreationPolicy.ACCEPT_ALL);
```

GetGameSessionId()

Recupera l'ID della sessione di gioco ospitata dal processo del server attivo.

Per i processi inattivi che non vengono attivati con una sessione di gioco, la chiamata restituisce un. [the section called “GameLiftError”](#)

Sintassi

```
AwsStringOutcome GetGameSessionId()
```

Valore restituito

Se l'esito è positivo, l'ID della sessione di gioco verrà restituito come oggetto [the section called “AwsStringOutcome”](#). Se non riesce, restituisce un messaggio di errore».

Esempio

```
AwsStringOutcome getGameSessionIdOutcome = GameLiftServerAPI.GetGameSessionId();
```

GetTerminationTime()

Restituisce il tempo di arresto pianificato di un processo del server, se è disponibile un tempo di chiusura. Un processo server esegue questa azione dopo aver ricevuto una `onProcessTerminate()` richiamata da Amazon GameLift Servers. Amazon GameLift Servers chiama `onProcessTerminate()` per i seguenti motivi:

- Quando il processo del server ha segnalato problemi di salute o non ha risposto. Amazon GameLift Servers
- Quando si termina l'istanza durante un evento di scale-down.
- [Quando un'istanza viene terminata a causa di un'interruzione di un'istanza spot.](#)

Sintassi

```
AwsDateTimeOutcome GetTerminationTime()
```

Valore restituito

In caso di successo, restituisce l'ora di terminazione come oggetto. [the section called “AwsDateTimeOutcome”](#) Il valore è il tempo di terminazione, espresso in segni di spunta trascorsi da allora. 0001 00:00:00 Ad esempio, il valore della data e dell'ora è uguale ai segni di

2020-09-13 12:26:40 -000Z spunta. 637355968000000000 Se non è disponibile alcun orario di terminazione, restituisce un messaggio di errore.

Esempio

```
AwsDateTimeOutcome getTerminationTimeOutcome = GameLiftServerAPI.GetTerminationTime();
```

AcceptPlayerSession()

Notifica Amazon GameLift Servers che un giocatore con l'ID di sessione specificato si è connesso al processo del server e deve essere convalidato. Amazon GameLift Servers verifica che l'ID della sessione del giocatore sia valido. Dopo la convalida della sessione di gioco, Amazon GameLift Servers modifica lo stato dello slot del giocatore da RISERVATO ad ATTIVO.

Sintassi

```
GenericOutcome AcceptPlayerSession(String playerId)
```

Parameters

playerSessionId

ID univoco rilasciato GameLift quando viene creata una nuova sessione giocatore.

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

Questo esempio illustra una funzione per la gestione di una richiesta di connessione, inclusa la convalida e il rifiuto di una sessione giocatore non valida. IDs

```
void ReceiveConnectingPlayerSessionID (Connection connection, String playerId)
{
    GenericOutcome acceptPlayerSessionOutcome =
    GameLiftServerAPI.AcceptPlayerSession(playerSessionId);
    if(acceptPlayerSessionOutcome.Success)
    {
        connectionToSessionMap.emplace(connection, playerId);
    }
}
```

```
        connection.Accept();
    }
    else
    {
        connection.Reject(acceptPlayerSessionOutcome.Error.ErrorMessage);
    }
}
```

RemovePlayerSession()

Notifica Amazon GameLift Servers che un giocatore si è disconnesso dal processo del server. In risposta, Amazon GameLift Servers modifica lo slot del giocatore rendendolo disponibile.

Sintassi

```
GenericOutcome RemovePlayerSession(String playerId)
```

Parameters

playerSessionId

ID univoco rilasciato Amazon GameLift Servers quando viene creata una nuova sessione giocatore.

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

```
GenericOutcome removePlayerSessionOutcome =
    GameLiftServerAPI.RemovePlayerSession(playerSessionId);
```

DescribePlayerSessions()

Recupera i dati della sessione del giocatore che includono impostazioni, metadati della sessione e dati del giocatore. Utilizza questa operazione per ottenere le informazioni per una singola sessione giocatore, per tutte le sessioni giocatore in una sessione di gioco o per tutte le sessioni giocatore associate a un singolo ID giocatore.

Sintassi

```
DescribePlayerSessionsOutcome DescribePlayerSessions(DescribePlayerSessionsRequest  
describePlayerSessionsRequest)
```

Parameters

[DescribePlayerSessionsRequest](#)

Un [the section called “DescribePlayerSessionsRequest”](#) oggetto che descrive le sessioni dei giocatori da recuperare.

Valore restituito

In caso di successo, restituisce un [the section called “DescribePlayerSessionsOutcome”](#) oggetto che contiene un set di oggetti della sessione del giocatore che soddisfano i parametri della richiesta.

Esempio

Questo esempio illustra una richiesta per tutte le sessioni giocatore attivamente connesse a una sessione di gioco specificata. Omettendo NextToken e impostando il valore Limite su 10, Amazon GameLift Servers restituirà i primi 10 record della sessione di gioco corrispondenti alla richiesta.

```
// Set request parameters  
DescribePlayerSessionsRequest describePlayerSessionsRequest = new  
    DescribePlayerSessionsRequest()  
{  
    GameSessionId = GameLiftServerAPI.GetGameSessionId().Result,    //gets the ID for the  
    current game session  
    Limit = 10,  
    PlayerSessionStatusFilter =  
        PlayerSessionStatusMapper.GetNameForPlayerSessionStatus(PlayerSessionStatus.ACTIVE)  
};  
// Call DescribePlayerSessions  
DescribePlayerSessionsOutcome describePlayerSessionsOutcome =  
    GameLiftServerAPI.DescribePlayerSessions(describePlayerSessionsRequest);
```

StartMatchBackfill()

Invia una richiesta per trovare nuovi giocatori per gli slot aperti in una sessione di gioco creata con FlexMatch. Per ulteriori informazioni, consulta la funzione di [FlexMatchriempimento](#).

Questa operazione è asincrona. Se vengono abbinati nuovi giocatori, Amazon GameLift Servers fornisce dati aggiornati sul matchmaker utilizzando la funzione di callback.

`OnUpdateGameSession()`

Un processo del server può avere un solo backfill degli abbinamenti attivo alla volta. Per inviare una nuova richiesta, chiama prima [StopMatchBackfill\(\)](#) per annullare la richiesta originale.

Sintassi

```
StartMatchBackfillOutcome StartMatchBackfill (StartMatchBackfillRequest  
startBackfillRequest);
```

Parameters

[StartMatchBackfillRequest](#)

Un `StartMatchBackfillRequest` oggetto contiene informazioni sulla richiesta di riempimento.

Valore restituito

Restituisce un [the section called "StartMatchBackfillOutcome"](#) oggetto con l'ID del ticket Match Backfill o restituisce un errore con un messaggio di errore.

Esempio

```
// Build a backfill request  
StartMatchBackfillRequest startBackfillRequest = new StartMatchBackfillRequest()  
{  
    TicketId = "1111aaaa-22bb-33cc-44dd-5555eeee66ff", //optional  
    MatchmakingConfigurationArn = "arn:aws:gamelift:us-  
west-2:111122223333:matchmakingconfiguration/MyMatchmakerConfig",  
    GameSessionId = GameLiftServerAPI.GetGameSessionId().Result, // gets ID for  
    current game session  
    MatchmakerData matchmakerData =  
        MatchmakerData.FromJson(gameSession.MatchmakerData), // gets matchmaker data for  
    current players  
    // get matchmakerData.Players  
    // remove data for players who are no longer connected  
    Players = ListOfPlayersRemainingInTheGame  
};  
  
// Send backfill request
```

```
StartMatchBackfillOutcome startBackfillOutcome =  
    GameLiftServerAPI.StartMatchBackfill(startBackfillRequest);  
  
// Implement callback function for backfill  
void OnUpdateGameSession(GameSession myGameSession)  
{  
    // game-specific tasks to prepare for the newly matched players and update matchmaker  
    data as needed  
}
```

StopMatchBackfill()

Annulla una richiesta di match backfill attiva. Per ulteriori informazioni, consulta la funzione di [FlexMatchriempimento](#).

Sintassi

```
GenericOutcome StopMatchBackfill (StopMatchBackfillRequest stopBackfillRequest);
```

Parameters

[StopMatchBackfillRequest](#)

Un `StopMatchBackfillRequest` oggetto che fornisce dettagli sul ticket di matchmaking che stai interrompendo.

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

```
// Set backfill stop request parameters  
StopMatchBackfillRequest stopBackfillRequest = new StopMatchBackfillRequest(){  
    TicketId = "1111aaaa-22bb-33cc-44dd-5555eeee66ff", //optional, if not provided one is  
    autogenerated  
    MatchmakingConfigurationArn = "arn:aws:gamelift:us-  
west-2:111122223333:matchmakingconfiguration/MyMatchmakerConfig",  
    GameSessionId = GameLiftServerAPI.GetGameSessionId().Result //gets the ID for the  
    current game session  
};
```

```
GenericOutcome stopBackfillOutcome =  
    GameLiftServerAPI.StopMatchBackfillRequest(stopBackfillRequest);
```

GetComputeCertificate()

Recupera il percorso del certificato TLS utilizzato per crittografare la connessione di rete tra il server di gioco e il client di gioco. Puoi utilizzare il percorso del certificato quando registri il tuo dispositivo di elaborazione in una flotta Anywhere. Amazon GameLift Servers Per ulteriori informazioni, consultare [RegisterCompute](#).

Sintassi

```
GetComputeCertificateOutcome GetComputeCertificate();
```

Valore restituito

Restituisce un `GetComputeCertificateResponse` oggetto che contiene quanto segue:

- `CertificatePath`: il percorso del certificato TLS sulla risorsa di calcolo. Quando si utilizza una flotta Amazon GameLift Servers gestita, questo percorso contiene:
 - `certificate.pem`: Il certificato dell'utente finale. La catena completa di certificati è la combinazione dei certificati `certificateChain.pem` aggiunti a questo certificato.
 - `certificateChain.pem`: La catena di certificati che contiene il certificato principale e i certificati intermedi.
 - `rootCertificate.pem`: Il certificato principale.
 - `privateKey.pem`: la chiave privata per il certificato dell'utente finale.
- `ComputeName`: il nome della risorsa di calcolo.

Esempio

```
GetComputeCertificateOutcome getComputeCertificateOutcome =  
    GameLiftServerAPI.GetComputeCertificate();
```

GetFleetRoleCredentials()

Recupera le credenziali del ruolo IAM che autorizzano Amazon GameLift Servers a interagire con altri. Servizi AWS Per ulteriori informazioni, consulta [Connect il server di gioco Amazon GameLift Servers ospitato ad altre AWS risorse](#).

Sintassi

```
GetFleetRoleCredentialsOutcome GetFleetRoleCredentials(GetFleetRoleCredentialsRequest request);
```

Parameters

[GetFleetRoleCredentialsRequest](#)

Credenziali di ruolo che estendono l'accesso limitato alle tue AWS risorse al server di gioco.

Valore restituito

Restituisce un oggetto [the section called “GetFleetRoleCredentialsOutcome”](#).

Esempio

```
// form the fleet credentials request
GetFleetRoleCredentialsRequest getFleetRoleCredentialsRequest = new
    GetFleetRoleCredentialsRequest(){
        RoleArn = "arn:aws:iam::123456789012:role/service-role/exampleGameLiftAction"
    };
GetFleetRoleCredentialsOutcome GetFleetRoleCredentialsOutcome credentials =
    GetFleetRoleCredentials(getFleetRoleCredentialsRequest);
```

Distruggi ()

Libera l'SDK del server di Amazon GameLift Servers gioco dalla memoria. È consigliabile chiamare questo metodo dopo `ProcessEnding()` e prima di terminare il processo. Se utilizzi una flotta Anywhere e non interrompi i processi del server dopo ogni sessione di gioco, chiama `Destroy()` e poi `InitSDK()` reinizializza prima di notificare Amazon GameLift Servers che il processo è pronto per ospitare una sessione di gioco. `ProcessReady()`

Sintassi

```
GenericOutcome Destroy()
```

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

```
// Operations to end game sessions and the server process
GenericOutcome processEndingOutcome = GameLiftServerAPI.ProcessEnding();

// Shut down and destroy the instance of the GameLift Game Server SDK
GenericOutcome destroyOutcome = GameLiftServerAPI.Destroy();

// Exit the process with success or failure
if (processEndingOutcome.Success)
{
    Environment.Exit(0);
}
else
{
    Console.WriteLine("ProcessEnding() failed. Error: " +
        processEndingOutcome.Error.ToString());
    Environment.Exit(-1);
}
```

Go server SDK per Amazon GameLift Servers -- Azioni

Usa il riferimento all'SDK 5.x del server per integrare il tuo gioco multiplayer con cui ospitarlo.

Amazon GameLift Servers Per indicazioni sul processo di integrazione, consulta [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#)

`GameLiftServerAPI` definisce le azioni dell'SDK del server Go.

Go server SDK per Amazon GameLift Servers: tipi di dati

Usa il riferimento all'SDK del server per integrare il tuo gioco multiplayer con cui ospitarlo. Amazon GameLift Servers Per indicazioni sul processo di integrazione, consulta [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#).

`GameLiftServerAPI` definisce le azioni dell'SDK del server Go.

[Go server SDK per Amazon GameLift Servers -- Azioni](#)

Tipi di dati

- [LogParameters](#)
- [MetricsParameters](#)

- [ProcessParameters](#)
- [UpdateGameSession](#)
- [GameSession](#)
- [ServerParameters](#)
- [StartMatchBackfillRequest](#)
- [Player](#)
- [DescribePlayerSessionsRequest](#)
- [StopMatchBackfillRequest](#)
- [GetFleetRoleCredentialsRequest](#)

LogParameters

Un oggetto che identifica i file generati durante una sessione di gioco che desideri caricare e archiviare Amazon GameLift Servers al termine della sessione di gioco. Il server di gioco LogParameters lo fornisce Amazon GameLift Servers come parte di un ProcessParameters oggetto in una [ProcessReady\(\)](#) chiamata.

Proprietà	Descrizione
LogPaths	L'elenco dei percorsi di directory dei file di registro del server di gioco che desideri Amazon GameLift Servers archiviare per accessi futuri. Il processo del server genera questi file durante ogni sessione di gioco. Definisci i percorsi e i nomi dei file nel tuo server di gioco e li memorizzi nella directory di build del gioco principale. I percorsi di registro devono essere assoluti. Ad esempio, se la build del gioco memorizza i log delle sessioni di gioco in un percorso come <code>MyGame\sessionLogs\</code>, il percorso si troverebbe <code>c:\game\MyGame\sessionLogs</code> su un'istanza di Windows. Tipo: <code>[]string</code> Obbligatorio: no

MetricsParameters

Un oggetto che comunica i parametri di configurazione per l'inizializzazione del sistema di metriche. Questa configurazione viene utilizzata per impostare la segnalazione StatSD, la segnalazione di arresti anomali e il comportamento di elaborazione delle metriche. Il server di gioco fornisce `MetricsParameters` to Amazon GameLift Servers come parte di una [InitMetrics\(\)](#) chiamata.

Proprietà	Descrizione
<code>StatsdHost</code>	L'host del server StatSD per la segnalazione delle metriche (ad esempio, «localhost»).
	Tipo: <code>string</code>
	Obbligatorio: sì
<code>StatsdPort</code>	La porta del server StatSD per la segnalazione delle metriche (ad esempio, 8125).
	Tipo: <code>int</code>
	Obbligatorio: sì
<code>CrashReporterHost</code>	L'host crash reporter per il tracciamento degli arresti anomali e il monitoraggio dei processi.
	Tipo: <code>string</code>
	Obbligatorio: sì
<code>CrashReporterPort</code>	La porta crash reporter per il tracciamento degli incidenti e il monitoraggio dei processi.
	Tipo: <code>int</code>
	Obbligatorio: sì
<code>FlushIntervalMs</code>	L'intervallo di flush delle metriche in millisecondi. Controlla la frequenza con cui le metriche vengono inviate a StatSD.
	Tipo: <code>int</code>

	Obbligatorio: sì
MaxPacketSize	La dimensione massima del pacchetto per le metriche in byte. Limita la dimensione dei pacchetti UDP inviati a StatSD.
	Tipo: <code>int</code>
	Obbligatorio: sì

ProcessParameters

Un oggetto che descrive la comunicazione tra un processo server e Amazon GameLift Servers. Il processo server fornisce queste informazioni a Amazon GameLift Servers con una chiamata a [ProcessReady\(\)](#).

Proprietà	Descrizione
LogParameters	Un oggetto con percorsi di directory ai file generati durante una sessione di gioco. Amazon GameLift Servers copia e archivia i file per accessi futuri. Tipo: LogParameters Obbligatorio: no
OnHealthCheck	La funzione di callback che Amazon GameLift Servers richiama per richiedere e un rapporto sullo stato di salute del processo del server. Amazon GameLift Servers chiama questa funzione ogni 60 secondi e attende una risposta per 60 secondi. Il processo del server ritorna <code>TRUE</code> se integro, <code>FALSE</code> se non integro. Se non viene restituita alcuna risposta, Amazon GameLift Servers registra il processo del server come non integro. Tipo: <code>OnHealthCheck func() bool</code> Obbligatorio: no
OnProcessTerminate	La funzione di callback che Amazon GameLift Servers richiama per forzare l'arresto del processo server. Dopo aver chiamato questa funzione, Amazon GameLift Servers attende 5 minuti che il processo server si spenga e risponda con una ProcessEnding() chiamata prima di chiudere il processo server.

	Tipo: <code>OnProcessTerminate func()</code> Obbligatorio: sì
<code>OnStartGameSession</code>	La funzione di callback che Amazon GameLift Servers richiama il passaggio di un oggetto di sessione di gioco aggiornato al processo del server. Amazon GameLift Servers chiama questa funzione quando è stata elaborata una richiesta di match backfill per fornire dati aggiornati sul matchmaker. Passa un GameSession oggetto, uno status update (<code>updateReason</code>) e l'ID del ticket match backfill. Tipo: <code>OnStartGameSession func (model.GameSession)</code> Obbligatorio: sì
<code>OnUpdateGameSession</code>	La funzione di callback che Amazon GameLift Servers richiama per passare informazioni aggiornate sulla sessione di gioco al processo del server. Amazon GameLift Servers chiama questa funzione dopo aver elaborato una richiesta di match backfill per fornire dati aggiornati sul matchmaker. Tipo: <code>OnUpdateGameSession func (model.UpdateGameSession)</code> Obbligatorio: no
<code>Port</code>	Il numero di porta su cui il processo del server ascolta le connessioni di nuovi giocatori. Il valore deve rientrare nella gamma di porte configurate per qualsiasi parco istanze che distribuisce questa build del server di gioco. Questo numero di porta è incluso nella sessione di gioco e negli oggetti della sessione del giocatore utilizzati dalle sessioni di gioco per la connessione a un processo server. Tipo: <code>int</code> Obbligatorio: sì

UpdateGameSession

Gli aggiornamenti a un oggetto della sessione di gioco, che include il motivo per cui la sessione di gioco è stata aggiornata e il relativo ID del ticket di backfill se il backfill viene utilizzato per riempire le sessioni dei giocatori nella sessione di gioco.

Properties	Descrizione
GameSession	Oggetto GameSession . L'GameSession oggetto contiene proprietà che descrivono una sessione di gioco. Tipo: GameSession GameSession() Obbligatorio: sì
UpdateReason	Il motivo per cui la sessione di gioco viene aggiornata. Tipo: UpdateReason UpdateReason() Obbligatorio: sì
BackfillTicketId	L'ID del ticket di backfill che tenta di aggiornare la sessione di gioco. Tipo: String Obbligatorio: no

GameSession

I dettagli di una sessione di gioco.

Properties	Descrizione
GameSessionId	Un identificatore univoco per la sessione di gioco. Una sessione di gioco Amazon Resource Name (ARN) ha il seguente formato: <code>arn:aws:gamelift:<region>::gamesession/<fleet ID>/<custom ID string or idempotency token></code> Tipo: String

Properties	Descrizione
	Obbligatorio: no
Name	Un'etichetta descrittiva della sessione di gioco. Tipo: <code>String</code> Obbligatorio: no
FleetId	Un identificatore univoco per la flotta su cui è in esecuzione la sessione di gioco. Tipo: <code>String</code> Obbligatorio: no
MaximumPlayerSessionCount	Il numero massimo di connessioni dei giocatori alla sessione di gioco. Tipo: <code>Integer</code> Obbligatorio: no
Porta	Il numero di porta per la sessione di gioco. Per connettersi a un server di Amazon GameLift Servers gioco, un'app necessita sia dell'indirizzo IP che del numero di porta. Tipo: <code>Integer</code> Obbligatorio: no
IpAddress	L'indirizzo IP della sessione di gioco. Per connettersi a un server di Amazon GameLift Servers gioco, un'app necessita sia dell'indirizzo IP che del numero di porta. Tipo: <code>String</code> Obbligatorio: no

Properties	Descrizione
GameSessionData	Un set di proprietà personalizzate della sessione di gioco, formattate come valore di stringa singola. Tipo: <code>String</code> Obbligatorio: no
MatchmakerData	Le informazioni sul processo di matchmaking utilizzato per creare la sessione di gioco, in sintassi JSON, sono formattate come stringa. Oltre alla configurazione di matchmaking utilizzata, contiene dati su tutti i giocatori assegnati alla partita, compresi gli attributi dei giocatori e le assegnazioni delle squadre. Tipo: <code>String</code> Obbligatorio: no
GameProperties	Un insieme di proprietà personalizzate per una sessione di gioco, formattate e come coppie chiave:valore. Queste proprietà vengono passate con una richiesta di avvio di una nuova sessione di gioco. Tipo: <code>map[string] string</code> Obbligatorio: no

Properties	Descrizione
DnsName	L'identificatore DNS assegnato all'istanza che esegue la sessione di gioco. I valori hanno il seguente formato: • Flotte abilitate per TLS: <code><unique identifier>.<region identifier>.amazongamelift.com</code> • Non-TLS-enabled flotte: <code>ec2-<unique identifier>.compute.amazonaws.com</code> Quando ti connetti a una sessione di gioco in esecuzione su una flotta abilitata per TLS, devi usare il nome DNS, non l'indirizzo IP. Tipo: String Obbligatorio: no

ServerParameters

Informazioni utilizzate per mantenere la connessione tra un server Amazon GameLift Servers Anywhere e il servizio. Amazon GameLift Servers Queste informazioni vengono utilizzate quando si avviano nuovi processi server con [InitSDK\(\)](#). Per i server ospitati su istanze EC2 Amazon GameLift Servers gestite, utilizza un oggetto vuoto.

Properties	Descrizione
WebSocket URL	GameLiftServerSdkEndpoint Amazon GameLift ServersRestituisce quando si utilizza una RegisterCompute risorsa di calcolo Amazon GameLift Servers Anywhere. Tipo: string Obbligatorio: sì
ProcessID	Un identificatore univoco registrato nel processo del server che ospita il gioco. Tipo: string

Properties	Descrizione
	Obbligatorio: sì
HostID	L'identificatore univoco della risorsa di elaborazione che ospita il nuovo processo del server. HostIDViene ComputeName utilizzato al momento della registrazione del computer. Per ulteriori informazioni, consulta RegisterCompute. Tipo: string Obbligatorio: sì
FleetID	L'identificatore univoco della flotta su cui è registrato il calcolo. Per ulteriori informazioni, consulta RegisterCompute. Tipo: string Obbligatorio: sì
AuthToken	Il token di autenticazione generato da Amazon GameLift Servers questo codice autentica il server. Amazon GameLift Servers Per ulteriori informazioni, consulta GetComputeAuthToken. Tipo: string Obbligatorio: sì

StartMatchBackfillRequest

Informazioni utilizzate per creare una richiesta di matchmaking backfill. Il server di gioco comunica queste informazioni durante una chiamata Amazon GameLift Servers. [StartMatchBackfill\(\)](#)

Properties	Descrizione
GameSessionArn	L'identificatore univoco della sessione di gioco. L'operazione API GetGameSessionId restituisce l'identificatore in formato ARN. Tipo: String

Properties	Descrizione
	Obbligatorio: sì
MatchmakingConfigurationArn	L'identificatore univoco (sotto forma di ARN) che il matchmaker deve utilizzare per questa richiesta. L'ARN del matchmaker per la sessione di gioco originale si trova nell'oggetto della sessione di gioco nella proprietà matchmaker data. Per ulteriori informazioni sui dati del matchmaker, vedi Lavora con i dati del matchmaker. Tipo: <code>String</code> Obbligatorio: sì
Players	Un insieme di dati che rappresenta tutti i giocatori che sono attualmente impegnati nella sessione di gioco. Il matchmaker utilizza queste informazioni per cercare nuovi giocatori che rappresentano un buon abbinamento per i giocatori attuali. Tipo: <code>[]model.Player</code> Obbligatorio: sì
TicketId	L'identificatore univoco di un ticket di richiesta di matchmaking o match backfill. Se non fornisci un valore, Amazon GameLift Servers ne genera uno. Utilizzare questo identificatore per monitorare lo stato del ticket di backfill degli abbinamenti o annullare la richiesta, se necessario. Tipo: <code>String</code> Obbligatorio: no

Player

L'oggetto che rappresenta un giocatore in matchmaking. Quando inizia una richiesta di matchmaking, un giocatore ha un ID giocatore, attributi e possibilmente dati sulla latenza. Amazon GameLift Servers aggiunge le informazioni sulla squadra dopo una partita.

Properties	Descrizione
LatencyInSM	Un insieme di valori espressi in millisecondi che indica la quantità di latenza che un giocatore sperimenta quando è connesso a una posizione. Se viene utilizzata questa proprietà, il giocatore viene abbinato solo per le posizioni elencate. Se un matchmaker ha una regola che valuta la latenza dei giocatori, i giocatori devono segnalare la latenza per essere abbinati. Tipo: <code>map[string] int</code> Obbligatorio: no
PlayerAttributes	Una raccolta di coppie chiave:valore che contengono informazioni sui giocatori da utilizzare nel matchmaking. Le chiavi degli attributi del giocatore devono corrispondere a quelle PlayerAttributes utilizzate in un set di regole di matchmaking. Per ulteriori informazioni sugli attributi dei giocatori, vedi AttributeValue. Tipo: <code>map[string] AttributeValue</code> Obbligatorio: no
PlayerId	Un identificatore univoco per un giocatore. Tipo: <code>String</code> Obbligatorio: no
Team	Il nome della squadra a cui è assegnato il giocatore in una partita. Definisci il nome della squadra nel set di regole del matchmaking. Tipo: <code>String</code> Obbligatorio: no

DescribePlayerSessionsRequest

Un oggetto che specifica le sessioni dei giocatori da recuperare. Il processo server fornisce queste informazioni con una [DescribePlayerSessions\(\)](#) chiamata a. Amazon GameLift Servers

Properties	Descrizione
GameSessionID	Un identificatore univoco della sessione di gioco. Utilizzare questo parametro per richiedere tutte le sessioni giocatore per la sessione di gioco specificata. Il formato dell'ID della sessione di gioco è <code>arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string></code>. GameSessionID È una stringa ID personalizzata o una stringa generata. Tipo: String Obbligatorio: no
PlayerSessionID	L'identificatore univoco per la sessione di un giocatore. Usa questo parametro per richiedere una singola sessione di gioco specifica. Tipo: String Obbligatorio: no
PlayerID	L'identificatore univoco di un giocatore. Usa questo parametro per richiedere tutte le sessioni di gioco per un giocatore specifico. Per informazioni, consulta Genera giocatore IDs. Tipo: String Obbligatorio: no
PlayerSessionStatusFilter	Lo stato della sessione del giocatore su cui filtrare i risultati. I possibili stati delle sessioni di gioco includono: • RISERVATO — La richiesta di sessione del giocatore è stata ricevuta, ma il giocatore non si è connesso al processo del server né è stato convalidato. • ATTIVO: il giocatore è stato convalidato dal processo del server ed è connesso.

Properties	Descrizione
	• COMPLETATA — La connessione del giocatore è stata interrotta. • TIMEDOUT: è stata ricevuta una richiesta di sessione, ma il giocatore non si è connesso o non è stato convalidato entro il limite di timeout (60 secondi). Tipo: String Obbligatorio: no
NextToken	Il token che indica l'inizio della pagina successiva di risultati. Per specificare l'inizio del set di risultati, non fornire un valore. Se fornisci un ID di sessione del giocatore, questo parametro viene ignorato. Tipo: String Obbligatorio: no
Limit	Numero massimo di risultati da restituire. Se fornisci un ID di sessione del giocatore, questo parametro viene ignorato. Tipo: int Obbligatorio: no

StopMatchBackfillRequest

Informazioni utilizzate per annullare una richiesta di matchmaking backfill. Il server di gioco comunica queste informazioni al Amazon GameLift Servers servizio durante una chiamata. [StopMatchBackfill\(\)](#)

Properties	Descrizione
GameSessionArn	L'identificatore univoco della sessione di gioco della richiesta che viene annullata. Tipo: string Obbligatorio: no

Properties	Descrizione
MatchmakingConfigurationArn	L'identificatore univoco del matchmaker a cui è stata inviata questa richiesta. Tipo: <code>string</code> Obbligatorio: no
TicketId	L'identificatore univoco del ticket di richiesta di riempimento da annullare. Tipo: <code>string</code> Obbligatorio: no

GetFleetRoleCredentialsRequest

Le credenziali del ruolo che estendono l'accesso limitato alle tue AWS risorse al server di gioco. Per ulteriori informazioni, consultare [Configurare un ruolo di servizio IAM per Amazon GameLift Servers](#).

Properties	Descrizione
RoleArn	L'ARN del ruolo di servizio che estende l'accesso limitato alle AWS risorse. Tipo: <code>string</code> Obbligatorio: sì
RoleSessionName	Il nome della sessione che descrive l'uso delle credenziali del ruolo. Tipo: <code>string</code> Obbligatorio: sì

[Go server SDK per Amazon GameLift Servers: tipi di dati](#)

Azioni

- [GetSdkVersion\(\)](#)
- [InitMetrics\(\)](#)
- [InitMetricsFromEnvironment\(\)](#)

- [InitSDK\(\)](#)
- [ProcessReady\(\)](#)
- [ProcessEnding\(\)](#)
- [ActivateGameSession\(\)](#)
- [UpdatePlayerSessionCreationPolicy\(\)](#)
- [GetGameSessionId\(\)](#)
- [GetTerminationTime\(\)](#)
- [AcceptPlayerSession\(\)](#)
- [RemovePlayerSession\(\)](#)
- [DescribePlayerSessions\(\)](#)
- [StartMatchBackfill\(\)](#)
- [StopMatchBackfill\(\)](#)
- [GetComputeCertificate\(\)](#)
- [GetFleetRoleCredentials\(\)](#)
- [Distruggi \(\)](#)

GetSdkVersion()

Restituisce il numero di versione corrente dell'SDK integrato nel processo del server.

Sintassi

```
func GetSdkVersion() (string, error)
```

Valore restituito

In caso di successo, restituisce la versione SDK corrente come stringa. La stringa restituita include il numero di versione (esempio `5.0.0`). Se non riesce, restituisce un messaggio di errore come `common.SdkVersionDetectionFailed`.

Esempio

```
version, err := server.GetSdkVersion()
```

InitMetrics()

Inizializza la raccolta di metriche per l'SDK. Amazon GameLift Servers Questo metodo imposta la reportistica delle metriche per aiutare a monitorare le prestazioni e lo stato del server. Chiama questo metodo dopo [InitSDK\(\)](#) ma prima [ProcessReady\(\)](#).

Sintassi

```
func InitMetrics() error
func InitMetrics(metricsParameters MetricsParameters) error
```

Parameters

MetricsParameters (facoltativo)

Un `MetricsParameters` oggetto che configura la raccolta di metriche. Se non viene fornita, viene utilizzata la configurazione delle metriche predefinita. La `MetricsParameters` struttura contiene i seguenti campi:

- `StatsdHost`- Il nome host o l'indirizzo IP del server StatSD.
- `StatsdPort`- Il numero di porta per il server StatSD.
- `CrashReporterHost`- Il nome host o l'indirizzo IP del servizio crash reporter.
- `CrashReporterPort`- Il numero di porta per il servizio crash reporter.
- `FlushIntervalMs`- L'intervallo in millisecondi per il lavaggio dei dati delle metriche.
- `MaxPacketSize`- La dimensione massima dei pacchetti di metriche in byte.

Per ulteriori informazioni sulla `MetricsParameters` struttura, consulta [Server SDK 5.x](#) per i tipi di dati C#.

Valore restituito

In caso di successo, restituisce `nil` un errore per indicare che la raccolta delle metriche è stata inizializzata correttamente.

Esempio

Inizializza le metriche con la configurazione predefinita:

```
err := server.InitMetrics()
```


Inizializza le metriche con una configurazione personalizzata:

```
metricsParams := MetricsParameters{
    StatsdHost:      "localhost",
    StatsdPort:      8125,
    CrashReporterHost: "localhost",
    CrashReporterPort: 9125,
    FlushIntervalMs:  5000,
    MaxPacketSize:    1024,
}

err := server.InitMetrics(metricsParams)
```

InitMetricsFromEnvironment()

Inizializza la raccolta di metriche per l'Amazon GameLift Servers SDK utilizzando la configurazione delle variabili di ambiente. Questo metodo configura il reporting delle metriche utilizzando le impostazioni predefinite derivate dall'ambiente di runtime.

Chiama questo metodo dopo [InitSDK\(\)](#) ma prima [ProcessReady\(\)](#).

Sintassi

```
func InitMetricsFromEnvironment() error
```

Valore restituito

In caso di successo, restituisce `nil` un errore per indicare che la raccolta delle metriche è stata inizializzata correttamente utilizzando la configurazione dell'ambiente.

Esempio

```
err := server.InitMetricsFromEnvironment()
```

InitSDK()

Inizializza l'SDK Amazon GameLift Servers. Chiama questo metodo all'avvio prima che si verifichi qualsiasi altra inizializzazione correlata a. Amazon GameLift Servers Questo metodo imposta la comunicazione tra il server e il Amazon GameLift Servers servizio.

Sintassi

```
func InitSDK(params ServerParameters) error
```

Parameters

ServerParameters

Per inizializzare un server di gioco su una flotta Amazon GameLift Servers Anywhere, costruisci un `ServerParameters` oggetto con le seguenti informazioni:

- L'URL WebSocket utilizzato per connettersi al server di gioco.
- L'ID del processo utilizzato per ospitare il server di gioco.
- L'ID del computer che ospita i processi del server di gioco.
- L'ID della Amazon GameLift Servers flotta contenente il tuo computer Amazon GameLift Servers Anywhere.
- Il token di autorizzazione generato dall'Amazon GameLift Servers operazione.

Per inizializzare un server di gioco su una flotta EC2 Amazon GameLift Servers gestita, costruisci un `ServerParameters` oggetto senza parametri. Con questa chiamata, l'Amazon GameLift Servers agente configura l'ambiente di calcolo e si connette automaticamente al Amazon GameLift Servers servizio per te.

Valore restituito

In caso di successo, restituisce un `nil` errore per indicare che il processo del server è pronto per la chiamata [ProcessReady\(\)](#).

Note

Se le chiamate a `InitSDK()` riescono per le build di gioco distribuite sulle flotte Anywhere, controlla il `ServerSdkVersion` parametro utilizzato durante la creazione della risorsa di compilazione. È necessario impostare esplicitamente questo valore sulla versione SDK del server in uso. Il valore predefinito per questo parametro è 4.x, che non è compatibile. Per risolvere questo problema, crea una nuova build e distribuiscila in una nuova flotta.

Esempio

Amazon GameLift ServersEsempio: Ovunque

```
//Define the server parameters
serverParameters := ServerParameters {
  WebSocketURL: "wss://us-west-1.api.amazongamelift.com",
  ProcessID: "PID1234",
  HostID: "HardwareAnywhere",
  FleetID: "aarn:aws:gamelift:us-west-1:111122223333:fleet/
fleet-9999ffff-88ee-77dd-66cc-5555bbbb44aa",
  AuthToken: "1111aaaa-22bb-33cc-44dd-5555eeee66ff"
}

//Call InitSDK to establish a local connection with the Amazon GameLift Servers Agent
to enable further communication.
err := server.InitSDK(serverParameters)
```

Amazon GameLift Serversesempio EC2 gestito

```
//Define the server parameters
serverParameters := ServerParameters {}

//Call InitSDK to establish a local connection with the Amazon GameLift Servers Agent
to enable further communication.
err := server.InitSDK(serverParameters)
```

ProcessReady()

Notifica Amazon GameLift Servers che il processo del server è pronto per ospitare sessioni di gioco. Chiama questo metodo dopo averlo [InitSDK\(\)](#) invocato. Questo metodo deve essere chiamato solo una volta per processo.

Sintassi

```
func ProcessReady(param ProcessParameters) error
```

Parameters

ProcessParameters

Un [ProcessParameters](#) oggetto comunica le seguenti informazioni sul processo del server:

- I nomi dei metodi di callback implementati nel codice del server di gioco richiamato dal Amazon GameLift Servers servizio per comunicare con il processo del server.
- Il numero di porta su cui il processo server è in ascolto.
- Il tipo di [LogParameters](#) dati contenente il percorso di tutti i file specifici della sessione di gioco che desideri acquisire e Amazon GameLift Servers archiviare.

Valore restituito

Restituisce un errore con un messaggio di errore se il metodo fallisce. Restituisce `nil` se il metodo ha successo.

Esempio

Questo esempio illustra la chiamata [ProcessReady\(\)](#) e le implementazioni della funzione delegata.

```
// Define the process parameters
processParams := ProcessParameters {
    OnStartGameSession: gameProcess.OnStartGameSession,
    OnUpdateGameSession: gameProcess.OnGameSessionUpdate,
    OnProcessTerminate: gameProcess.OnProcessTerminate,
    OnHealthCheck: gameProcess.OnHealthCheck,
    Port: port,
    LogParameters: LogParameters {    // logging and error example
        []string {"C:\\game\\logs", "C:\\game\\error"}
    }
}

err := server.ProcessReady(processParams)
```

ProcessEnding()

Notifica Amazon GameLift Servers che il processo del server sta terminando. Chiama questo metodo dopo tutte le altre attività di pulizia (inclusa la chiusura della sessione di gioco attiva) e prima di terminare il processo. A seconda del risultato di `ProcessEnding()`, il processo termina con successo (0) o errore (-1) e genera un evento relativo alla flotta. Se il processo termina con un errore, l'evento fleet generato è `SERVER_PROCESS_TERMINATED_UNHEALTHY`.

Sintassi

```
func ProcessEnding() error
```

Valore restituito

Restituisce un codice di errore 0 o un codice di errore definito.

Esempio

```
// operations to end game sessions and the server process
defer func() {
    err := server.ProcessEnding()
    server.Destroy()
    if err != nil {
        fmt.Println("ProcessEnding() failed. Error: ", err)
        os.Exit(-1)
    } else {
        os.Exit(0)
    }
}
```

ActivateGameSession()

Notifica Amazon GameLift Servers che il processo del server ha attivato una sessione di gioco ed è ora pronto per ricevere le connessioni dei giocatori. Questa azione viene richiamata come parte della funzione di `onStartGameSession()` callback, dopo l'inizializzazione di tutta la sessione di gioco.

Sintassi

```
func ActivateGameSession() error
```

Valore restituito

Restituisce un errore con un messaggio di errore se il metodo fallisce.

Esempio

Questo esempio mostra le `ActivateGameSession()` chiamate come parte della funzione `onStartGameSession()` delegata.

```
func OnStartGameSession(GameSession gameSession) {
    // game-specific tasks when starting a new game session, such as loading map
    // Activate when ready to receive players
    err := server.ActivateGameSession();
}
```

```
}
```

UpdatePlayerSessionCreationPolicy()

Aggiorna la capacità della sessione di gioco corrente di accettare nuove sessioni giocatore. Una sessione di gioco può essere configurata per accettare o rifiutare tutte le nuove sessioni giocatore.

Sintassi

```
func UpdatePlayerSessionCreationPolicy(policy model.PlayerSessionCreationPolicy) error
```

Parameters

playerSessionCreationPolitica

Valore di stringa che indica se la sessione di gioco accetta nuovi giocatori.

I valori validi includono:

- **model.AcceptAll**— Accetta tutte le sessioni di nuovi giocatori.
- **model.DenyAll**— Nega tutte le sessioni dei nuovi giocatori.

Valore restituito

Restituisce un errore con un messaggio di errore se si verifica un errore.

Esempio

Questo esempio definisce la policy di partecipazione alla sessione di gioco corrente per accettare tutti i giocatori.

```
err := server.UpdatePlayerSessionCreationPolicy(model.AcceptAll)
```

GetGameSessionId()

Recupera l'ID della sessione di gioco ospitata dal processo del server attivo.

Sintassi

```
func GetGameSessionID() (string, error)
```

Parameters

Questa operazione non prevede parametri.

Valore restituito

In caso di successo, restituisce l'ID della sessione di gioco e l'errore zero. Per i processi inattivi che non sono ancora stati attivati con una sessione di gioco, la chiamata restituisce una stringa vuota e `nil` un errore.

Esempio

```
gameSessionID, err := server.GetGameSessionID()
```

GetTerminationTime()

Restituisce l'ora in cui è pianificata la chiusura di un processo del server, se è disponibile un orario di terminazione. Un processo server esegue questa azione dopo aver ricevuto una `onProcessTerminate()` richiamata da Amazon GameLift Servers. Amazon GameLift Servers chiama `onProcessTerminate()` per i seguenti motivi:

- Quando il processo del server ha segnalato problemi di salute o non ha risposto. Amazon GameLift Servers
- Quando si termina l'istanza durante un evento di scale-down.
- [Quando un'istanza viene terminata a causa di un'interruzione di un'istanza spot.](#)

Sintassi

```
func GetTerminationTime() (int64, error)
```

Valore restituito

In caso di successo, restituisce il timestamp in secondi di epoca in cui è pianificata la chiusura del processo del server e l'interruzione di un errore. `nil` Il valore è il tempo di terminazione, espresso in tick trascorsi da `0001 00:00:00`. Ad esempio, il valore di data e ora è uguale ai segni di `2020-09-13 12:26:40 -0000Z` spunta. `6373559680000000000` Se non è disponibile alcun orario di terminazione, restituisce un messaggio di errore.

Esempio

```
terminationTime, err := server.GetTerminationTime()
```

AcceptPlayerSession()

Notifica Amazon GameLift Servers che un giocatore con l'ID di sessione specificato si è connesso al processo del server e deve essere convalidato. Amazon GameLift Servers verifica che l'ID della sessione del giocatore sia valido. Dopo la convalida della sessione del giocatore, Amazon GameLift Servers modifica lo stato dello slot del giocatore da `aRESERVED` a `ACTIVE`.

Sintassi

```
func AcceptPlayerSession(playerSessionID string) error
```

Parameters

playerSessionId

ID univoco rilasciato da Amazon GameLift Servers quando viene creata una nuova sessione giocatore.

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

Questo esempio gestisce una richiesta di connessione che include la convalida e il rifiuto di una sessione giocatore non valida. IDs

```
func ReceiveConnectingPlayerSessionID(conn Connection, playerSessionID string) {  
    err := server.AcceptPlayerSession(playerSessionID)  
    if err != nil {  
        connection.Accept()  
    } else {  
        connection.Reject(err.Error())  
    }  
}
```


RemovePlayerSession()

Notifica Amazon GameLift Servers che un giocatore si è disconnesso dal processo del server. In risposta, Amazon GameLift Servers modifica lo slot del giocatore rendendolo disponibile.

Sintassi

```
func RemovePlayerSession(playerSessionID string) error
```

Parameters

playerSessionId

ID univoco rilasciato Amazon GameLift Servers quando viene creata una nuova sessione giocatore.

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

```
err := server.RemovePlayerSession(playerSessionID)
```

DescribePlayerSessions()

Recupera i dati della sessione del giocatore che includono impostazioni, metadati della sessione e dati del giocatore. Utilizzate questo metodo per ottenere informazioni su quanto segue:

- Una sessione per giocatore singolo
- Tutte le sessioni dei giocatori in una sessione di gioco
- Tutte le sessioni dei giocatori sono associate a un ID giocatore singolo

Sintassi

```
func DescribePlayerSessions(req request.DescribePlayerSessionsRequest)  
(result.DescribePlayerSessionsResult, error) {  
    return srv.describePlayerSessions(&req)
```

```
}
```

Parameters

[DescribePlayerSessionsRequest](#)

Un `DescribePlayerSessionsRequest` oggetto descrive le sessioni dei giocatori da recuperare.

Valore restituito

In caso di successo, restituisce un `DescribePlayerSessionsResult` oggetto che contiene un set di oggetti della sessione del giocatore che soddisfano i parametri della richiesta.

Esempio

Questo esempio richiede che tutte le sessioni dei giocatori siano connesse attivamente a una sessione di gioco specificata. Omettendo `NextToken` impostando il valore `Limite` su 10, Amazon GameLift Servers restituisce i primi 10 record di sessione tra giocatori corrispondenti alla richiesta.

```
// create request
describePlayerSessionsRequest := request.NewDescribePlayerSessions()
describePlayerSessionsRequest.GameSessionID, _ = server.GetGameSessionID() // get ID
for the current game session
describePlayerSessionsRequest.Limit = 10 // return the
first 10 player sessions
describePlayerSessionsRequest.PlayerSessionStatusFilter = "ACTIVE" // Get all
player sessions actively connected to the game session

describePlayerSessionsResult, err :=
server.DescribePlayerSessions(describePlayerSessionsRequest)
```

StartMatchBackfill()

Invia una richiesta per trovare nuovi giocatori per gli slot aperti in una sessione di gioco creata con `FlexMatch`. Per ulteriori informazioni, consulta la funzione di [FlexMatchriempimento](#).

Questa operazione è asincrona. Se vengono abbinati nuovi giocatori, Amazon GameLift Servers fornisce dati aggiornati sul matchmaker utilizzando la funzione di callback.

OnUpdateGameSession()

Un processo del server può avere un solo backfill degli abbinamenti attivo alla volta. Per inviare una nuova richiesta, chiama prima [StopMatchBackfill\(\)](#) per annullare la richiesta originale.

Sintassi

```
func StartMatchBackfill(req request.StartMatchBackfillRequest)
    (result.StartMatchBackfillResult, error)
```

Parameters

[StartMatchBackfillRequest](#)

Un StartMatchBackfillRequest oggetto comunica le seguenti informazioni:

- ID del ticket da assegnare alla richiesta di backfill. Queste informazioni sono facoltative; se non viene fornito alcun ID, ne Amazon GameLift Servers genera uno.
- Matchmaker a cui inviare la richiesta. L'ARN di configurazione completo è obbligatorio. Questo valore si trova nei dati del matchmaker della sessione di gioco.
- L'ID della sessione di gioco da riempire.
- I dati di matchmaking disponibili per i giocatori attuali della sessione di gioco.

Valore restituito

Restituisce un StartMatchBackfillResult oggetto con l'ID del ticket match backfill, oppure restituisce un errore con un messaggio di errore.

Esempio

```
// form the request
startBackfillRequest := request.NewStartMatchBackfill()
startBackfillRequest.RequestID = "1111aaaa-22bb-33cc-44dd-5555eeee66ff" // optional
startBackfillRequest.MatchmakingConfigurationArn = "arn:aws:gamelift:us-west-2:111122223333:matchmakingconfiguration/MyMatchmakerConfig"
var matchMaker model.MatchmakerData
if err := matchMaker.UnmarshalJSON([]byte(gameSession.MatchmakerData)); err != nil {
    return
}
startBackfillRequest.Players = matchMaker.Players
```

```
res, err := server.StartMatchBackfill(startBackfillRequest)

// Implement callback function for backfill
func OnUpdateGameSession(myGameSession model.GameSession) {
    // game-specific tasks to prepare for the newly matched players and update
    matchmaker data as needed
}
```

StopMatchBackfill()

Annulla una richiesta di match backfill attiva. Per ulteriori informazioni, consulta la funzione di [FlexMatchriempimento](#).

Sintassi

```
func StopMatchBackfill(req request.StopMatchBackfillRequest) error
```

Parameters

[StopMatchBackfillRequest](#)

Un StopMatchBackfillRequest oggetto che identifica il ticket di matchmaking da annullare:

- L'ID del ticket assegnato alla richiesta di riempimento.
- Il matchmaker a cui è stata inviata la richiesta di riempimento.
- La sessione di gioco associata alla richiesta di riempimento.

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

```
stopBackfillRequest := request.NewStopMatchBackfill() // Use this function to create
request
stopBackfillRequest.TicketID = "1111aaaa-22bb-33cc-44dd-5555eeee66ff"
stopBackfillRequest.MatchmakingConfigurationArn = "arn:aws:gamelift:us-
west-2:111122223333:matchmakingconfiguration/MyMatchmakerConfig"
```

```
//error
err := server.StopMatchBackfill(stopBackfillRequest)
```

GetComputeCertificate()

Recupera il percorso del certificato TLS utilizzato per crittografare la connessione di rete tra il server di gioco e il client di gioco. Puoi utilizzare il percorso del certificato quando registri il tuo dispositivo di elaborazione in una flotta Anywhere. Amazon GameLift Servers Per ulteriori informazioni, consulta [RegisterCompute](#).

Sintassi

```
func GetComputeCertificate() (result.GetComputeCertificateResult, error)
```

Valore restituito

Restituisce un `GetComputeCertificateResult` oggetto che contiene quanto segue:

- `CertificatePath`: il percorso del certificato TLS sulla risorsa di calcolo. Quando si utilizza una flotta Amazon GameLift Servers gestita, questo percorso contiene:
 - `certificate.pem`: Il certificato dell'utente finale. La catena completa di certificati è la combinazione dei certificati `certificateChain.pem` aggiunti a questo certificato.
 - `certificateChain.pem`: La catena di certificati che contiene il certificato principale e i certificati intermedi.
 - `rootCertificate.pem`: Il certificato principale.
 - `privateKey.pem`: la chiave privata per il certificato dell'utente finale.
- `ComputeName`: il nome della risorsa di calcolo.

Esempio

```
tlsCertificate, err := server.GetFleetRoleCredentials(getFleetRoleCredentialsRequest)
```

GetFleetRoleCredentials()

Recupera le credenziali del ruolo di servizio che crei per estendere le autorizzazioni agli altri utenti. Servizi AWS Amazon GameLift Servers Queste credenziali consentono al server di gioco di utilizzare

le tue risorse. AWS Per ulteriori informazioni, consulta [Configurare un ruolo di servizio IAM per Amazon GameLift Servers](#).

Sintassi

```
func GetFleetRoleCredentials(  
    req request.GetFleetRoleCredentialsRequest,  
) (result.GetFleetRoleCredentialsResult, error) {  
    return srv.getFleetRoleCredentials(&req)  
}
```

Parameters

[GetFleetRoleCredentialsRequest](#)

Credenziali di ruolo che estendono l'accesso limitato alle tue AWS risorse al server di gioco.

Valore restituito

Restituisce un `GetFleetRoleCredentialsResult` oggetto che contiene quanto segue:

- `AssumedRoleUserArn` - L'Amazon Resource Name (ARN) dell'utente a cui appartiene il ruolo di servizio.
- `AssumedRoleId` - L'ID dell'utente a cui appartiene il ruolo di servizio.
- `AccessKeyId` - L'ID della chiave di accesso per autenticare e fornire l'accesso alle AWS risorse.
- `SecretAccessKey` - L'ID della chiave di accesso segreta per l'autenticazione.
- `SessionToken` - Un token per identificare la sessione attiva corrente che interagisce con AWS le tue risorse.
- `Scadenza`: il periodo di tempo che manca alla scadenza delle credenziali della sessione.

Esempio

```
// form the customer credentials request  
getFleetRoleCredentialsRequest := request.NewGetFleetRoleCredentials()  
getFleetRoleCredentialsRequest.RoleArn = "arn:aws:iam::123456789012:role/service-role/  
exampleGameLiftAction"  
  
credentials, err := server.GetFleetRoleCredentials(getFleetRoleCredentialsRequest)
```

Distruggi ()

Libera l'SDK del server di Amazon GameLift Servers gioco dalla memoria. È consigliabile chiamare questo metodo dopo `ProcessEnding()` e prima di terminare il processo. Se utilizzi una flotta Anywhere e non interrompi i processi del server dopo ogni sessione di gioco, chiama `Destroy()` e poi `InitSDK()` reinizializza prima di notificare Amazon GameLift Servers che il processo è pronto per ospitare una sessione di gioco. `ProcessReady()`

Sintassi

```
func Destroy() error {  
    return srv.destroy()  
}
```

Valore restituito

Restituisce un errore con un messaggio di errore se il metodo fallisce.

Esempio

```
// operations to end game sessions and the server process  
defer func() {  
    err := server.ProcessEnding()  
    server.Destroy()  
    if err != nil {  
        fmt.Println("ProcessEnding() failed. Error: ", err)  
        os.Exit(-1)  
    } else {  
        os.Exit(0)  
    }  
}
```

Server C++ (Unreal) SDK 5.x per -- Azioni Amazon GameLift Servers

Usa il riferimento all'Amazon GameLift ServersUnreal Server SDK 5.x per aiutarti a preparare il tuo gioco multiplayer da utilizzare con. Amazon GameLift Servers Per i dettagli sul processo di integrazione, consulta. [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#) Se stai usando il Amazon GameLift Servers plugin per Unreal, vedi anche [Amazon GameLift Serversplugin per Unreal Engine](#).

 Note

Questo argomento descrive l'API Amazon GameLift Servers C++ che puoi usare quando compili per Unreal Engine. In particolare, questa documentazione si applica al codice compilato con l'opzione. `-DBUILD_FOR_UNREAL=1`

Server C++ (Unreal) SDK 5.x per Amazon GameLift Servers -- Tipi di dati

Usa il riferimento all'Amazon GameLift Servers Unreal Server SDK 5.x per aiutarti a preparare il tuo gioco multiplayer da utilizzare con. Amazon GameLift Servers Per i dettagli sul processo di integrazione, consulta. [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#) Se stai usando il Amazon GameLift Servers plugin per Unreal, vedi anche [Amazon GameLift Servers plugin per Unreal Engine](#).

 Note

Questo argomento descrive l'API Amazon GameLift Servers C++ che puoi usare quando compili per Unreal Engine. In particolare, questa documentazione si applica al codice compilato con l'opzione. `-DBUILD_FOR_UNREAL=1`

[Server C++ \(Unreal\) SDK 5.x per -- Azioni Amazon GameLift Servers](#)

Tipi di dati

- [FProcessParametri](#)
- [UpdateGameSession](#)
- [GameSession](#)
- [FServerParametri](#)
- [FStartMatchBackfillRequest](#)
- [FPlayer](#)
- [FGameLiftDescribePlayerSessionsRequest](#)
- [FStopMatchBackfillRequest](#)
- [FAttributeValore](#)
- [FGameLiftGetFleetRoleCredentialsRequest](#)
- [FGameLiftLongOutcome](#)

- [FGameLiftStringOutcome](#)
- [FGameLiftDescribePlayerSessionsOutcome](#)
- [FGameLiftDescribePlayerSessionsResult](#)
- [FGenericRisultato](#)
- [FGameLiftPlayerSession](#)
- [FGameLiftGetComputeCertificateOutcome](#)
- [FGameLiftGetComputeCertificateResult](#)
- [FGameLiftGetFleetRoleCredentialsOutcome](#)
- [FGetFleetRoleCredentialsResult](#)
- [FGameLiftError](#)
- [enumerazioni;](#)

FProcessParametri

Questo tipo di dati contiene l'insieme di parametri inviati a Amazon GameLift Servers in [aProcessReady\(\)](#).

Proprietà	Descrizione
LogParameters	Un oggetto con percorsi di directory ai file generati durante una sessione di gioco. Amazon GameLift Servers copia e archivia i file per accessi futuri. Tipo: TArray<FString> Obbligatorio: no
OnHealthCheck	La funzione di callback che Amazon GameLift Servers richiama per richiedere un rapporto sullo stato di salute del processo del server. Amazon GameLift Servers chiama questa funzione ogni 60 secondi e attende una risposta per 60 secondi. Il processo del server ritorna TRUE se integro, FALSE se non integro. Se non viene restituita alcuna risposta, Amazon

GameLift Servers registra il processo del server come non integro.

Questa proprietà è una funzione delegata definita come `DECLARE_DELEGATE_RetVal(bool, FOnHealthCheck)` ;

Tipo: `FOnHealthCheck`

Obbligatorio: no

OnProcessTerminate

La funzione di callback che Amazon GameLift Servers richiama per forzare l'arresto del processo server. Dopo aver chiamato questa funzione, Amazon GameLift Servers attende 5 minuti che il processo server si spenga e risponda con una [ProcessEnding\(\)](#) chiamata prima di chiudere il processo server.

Tipo: `FSimpleDelegate`

Obbligatorio: sì

OnStartGameSession

La funzione di callback che Amazon GameLift Servers richiama l'attivazione di una nuova sessione di gioco. Amazon GameLift Servers chiama questa funzione in risposta a una richiesta del client. [CreateGameSession](#) La funzione di callback passa un [GameSession](#) oggetto.

Questa proprietà è una funzione delegata definita come `DECLARE_DELEGATE_OneParam(FOnStartGameSession, Aws::GameLift::Server::Model::GameSession)`;

Tipo: `FOnStartGameSession`

Obbligatorio: sì

OnUpdateGameSession

La funzione di callback che Amazon GameLift Servers richiama per passare un oggetto di sessione di gioco aggiornato al processo del server. Amazon GameLift Servers chiama questa funzione quando è stata elaborata una richiesta di match backfill per fornire dati aggiornati sul matchmaker. Passa un [GameSession](#) oggetto, uno status update (updateReason) e l'ID del ticket match backfill.

Questa proprietà è una funzione delegata definita come DECLARE_DELEGATE_OneParam(FOnUpdateGameSession, Aws::GameLift::Server::Model::UpdateGameSession);

Tipo: FOnUpdateGameSession

Obbligatorio: no

Porta

Il numero di porta su cui il processo server ascolta le connessioni di nuovi giocatori . Il valore deve rientrare nella gamma di porte configurate per qualsiasi parco istanze che distribuisce questa build del server di gioco. Questo numero di porta è incluso nella sessione di gioco e negli oggetti della sessione del giocatore utilizzati dalle sessioni di gioco per la connessione a un processo server.

Tipo: int

Obbligatorio: sì

UpdateGameSession

Questo tipo di dati viene aggiornato a un oggetto della sessione di gioco, che include il motivo per cui la sessione di gioco è stata aggiornata e il relativo ID del ticket di backfill se il backfill viene utilizzato per riempire le sessioni dei giocatori nella sessione di gioco.

Properties	Descrizione
GameSession	Oggetto GameSession. L'GameSession oggetto contiene proprietà che descrivono una sessione di gioco. Tipo: <code>Aws::GameLift::Server::GameSession</code> Obbligatorio: no
UpdateReason	Il motivo per cui la sessione di gioco viene aggiornata. Tipo: <code>enum class UpdateReason</code> • <code>MATCHMAKING_DATA_UPDATED</code> • <code>BACKFILL_FAILED</code> • <code>BACKFILL_TIMED_OUT</code> • <code>BACKFILL_ANNULLATO</code> Obbligatorio: no
BackfillTicketId	L'ID del ticket di backfill che tenta di aggiornare la sessione di gioco. Tipo: <code>char[]</code> Obbligatorio: no

GameSession

Questo tipo di dati fornisce i dettagli di una sessione di gioco.

Properties	Descrizione
GameSessionId	Un identificatore univoco per la sessione di gioco. L'ARN di una sessione di gioco ha il seguente formato: <code>arn:aws:gamelift:<region>::gamesession/<fleet ID>/<custom ID string or idempotency token></code> Tipo: <code>char[]</code> Obbligatorio: no
Name	Un'etichetta descrittiva della sessione di gioco. Tipo: <code>char[]</code> Obbligatorio: no
FleetId	Un identificatore univoco per la flotta su cui è in esecuzione la sessione di gioco. Tipo: <code>char[]</code> Obbligatorio: no
MaximumPlayerSessionCount	Il numero massimo di connessioni dei giocatori alla sessione di gioco. Tipo: <code>int</code> Obbligatorio: no
Porta	Il numero di porta per la sessione di gioco. Per connettersi a un server di Amazon GameLift Servers gioco, un'app necessita sia dell'indirizzo IP che del numero di porta. Tipo: <code>int</code>

Properties	Descrizione
	Obbligatorio: no
IpAddress	L'indirizzo IP della sessione di gioco. Per connettersi a un server di Amazon GameLift Servers gioco, un'app necessita sia dell'indirizzo IP che del numero di porta. Tipo: <code>char[]</code> Obbligatorio: no
GameSessionData	Un set di proprietà personalizzate della sessione di gioco, formattate come valore di stringa singola. Tipo: <code>char[]</code> Obbligatorio: no
MatchmakerData	Informazioni sul processo di matchmaking utilizzato per creare la sessione di gioco, in sintassi JSON, formattate come stringa. Oltre alla configurazione di matchmaking utilizzata, contiene dati su tutti i giocatori assegnati alla partita, compresi gli attributi dei giocatori e le assegnazioni delle squadre. Tipo: <code>char[]</code> Obbligatorio: no

Properties	Descrizione
GameProperties	Un insieme di proprietà personalizzate per una sessione di gioco, formattate come coppie chiave:valore. Queste proprietà vengono passate con una richiesta di avvio di una nuova sessione di gioco. Tipo: <code>GameProperty[]</code> Obbligatorio: no
DnsName	L'identificatore DNS assegnato all'istanza che esegue la sessione di gioco. I valori hanno il seguente formato: • Flotte abilitate per TLS: <code><unique identifier>.<region identifier>.amazongamelift.com</code> • Non-TLS-enabled flotte: <code>ec2-<unique identifier>.compute.amazonaws.com</code> Quando ti connetti a una sessione di gioco in esecuzione su una flotta abilitata per TLS, devi usare il nome DNS, non l'indirizzo IP. Tipo: <code>char[]</code> Obbligatorio: no

FServerParametri

Informazioni utilizzate per mantenere la connessione tra un server Amazon GameLift Servers Anywhere e il Amazon GameLift Servers servizio. Queste informazioni vengono utilizzate quando si avviano nuovi processi server con [InitSDK\(\)](#). Per i server ospitati su istanze EC2 Amazon GameLift Servers gestite, utilizza un oggetto vuoto.

Properties	Descrizione
websocketUrl	GameLiftServerSdkEndpoint Amazon GameLift ServersRestituisce quando si utilizza una RegisterCompute risorsa di calcolo Amazon GameLift Servers Anywhere. Tipo: <code>char[]</code> Obbligatorio: sì
ID del processo	Un identificatore univoco registrato nel processo del server che ospita il gioco. Tipo: <code>char[]</code> Obbligatorio: sì
hostId	HostIDViene ComputeName utilizzato al momento della registrazione del computer. Per ulteriori informazioni, consultare RegisterCompute. Tipo: <code>char[]</code> Obbligatorio: sì
FleetID	L'identificatore univoco della flotta su cui è registrato il computer. Per ulteriori informazioni, consultare RegisterCompute. Tipo: <code>char[]</code> Obbligatorio: sì
AuthToken	Il token di autenticazione generato da Amazon GameLift Servers questo autentica il server su. Amazon GameLift Servers Per ulteriori informazioni, consultare GetComputeAuthToken.

Properties	Descrizione
	Tipo: <code>char[]</code>
	Obbligatorio: sì

FStartMatchBackfillRequest

Informazioni utilizzate per creare una richiesta di matchmaking backfill. Il server di gioco comunica queste informazioni durante una chiamata Amazon GameLift Servers. [StartMatchBackfill\(\)](#)

Properties	Descrizione
GameSessionArn	Un identificatore univoco della sessione di gioco. L'operazione API GetGameSessionId restituisce l'identificatore in formato ARN. Tipo: <code>char[]</code> Obbligatorio: sì
MatchmakingConfigurationArn	Un identificatore univoco, sotto forma di ARN, che il matchmaker può utilizzare per questa richiesta. L'ARN del matchmaker per la sessione di gioco originale si trova nell'oggetto della sessione di gioco nella proprietà matchmaker data. Scopri di più sui dati dei matchmaker in Work with matchmaker data. Tipo: <code>char[]</code> Obbligatorio: sì
Players	Un insieme di dati che rappresenta tutti i giocatori che partecipano alla sessione di gioco. Il matchmaker utilizza queste informazioni per cercare nuovi giocatori che rappresentino un buon abbinamento per i giocatori attuali.

Properties	Descrizione
	Tipo: TArray<FPlayer> Obbligatorio: sì
TicketId	Un identificatore univoco per un ticket di richiesta di matchmaking o match backfill. Se non fornisci un valore, Amazon GameLift Servers ne genera uno. Utilizzare questo identificatore per monitorare lo stato del ticket di backfill degli abbinamenti o annullare la richiesta, se necessario. Tipo: char[] Obbligatorio: no

FPlayer

Questo tipo di dati rappresenta un giocatore in matchmaking. Quando si avvia una richiesta di matchmaking, un giocatore ha un ID giocatore, attributi e possibilmente dati di latenza. Amazon GameLift Servers aggiunge le informazioni sulla squadra dopo la fine di una partita.

Properties	Descrizione
LatencyInSM	Un insieme di valori espressi in millisecondi che indica la quantità di latenza che un giocatore sperimenta quando è connesso a una posizione. Se viene utilizzata questa proprietà, il giocatore viene abbinato solo per le posizioni elencate. Se un matchmaker ha una regola che valuta la latenza dei giocatori, i giocatori devono segnalare la latenza per essere abbinati. Tipo: TMap>FString, int32<

Properties	Descrizione
	Obbligatorio: no
PlayerAttributes	Una raccolta di coppie chiave:valore contenent i informazioni sui giocatori da utilizzare nel matchmaking. Le chiavi degli attributi del giocatore devono corrispondere a quelle PlayerAttributes utilizzate in un set di regole di matchmaking. Per ulteriori informazioni sugli attributi dei giocatori, vedi AttributeValue. Tipo: TMap>FString, FAttributeValue< Obbligatorio: no
PlayerId	Un identificatore univoco per un giocatore. Tipo: std::string Obbligatorio: no
Team	Il nome della squadra a cui è assegnato il giocatore in una partita. Definisci il nome della squadra nel set di regole del matchmaking. Tipo: FString Obbligatorio: no

FGameLiftDescribePlayerSessionsRequest

Un oggetto che specifica le sessioni dei giocatori da recuperare. Il processo server fornisce queste informazioni con una [DescribePlayerSessions\(\)](#) chiamata a. Amazon GameLift Servers

Properties	Descrizione
GameSessionId	Un identificatore univoco della sessione di gioco. Utilizzare questo parametro per richiedere e tutte le sessioni giocatore per la sessione di gioco specificata. Il formato dell'ID della sessione di gioco è FString. GameSessionID È una stringa ID personalizzata o un Tipo: std::string Obbligatorio: no
PlayerSessionId	L'identificatore univoco per la sessione di un giocatore. Usa questo parametro per richiedere una singola sessione di gioco specifica. Tipo: FString Obbligatorio: no
PlayerId	L'identificatore univoco di un giocatore. Usa questo parametro per richiedere tutte le sessioni di gioco per un giocatore specifico. Per informazioni, consulta Genera giocatore IDs. Tipo: FString Obbligatorio: no
PlayerSessionStatusFilter	Lo stato della sessione del giocatore su cui filtrare i risultati. I possibili stati delle sessioni di gioco includono: • RISERVATO — La richiesta di sessione del giocatore è stata ricevuta, ma il giocatore non si è connesso al processo del server né è stato convalidato.

Properties	Descrizione
	• ATTIVO: il giocatore è stato convalidato dal processo del server ed è connesso. • COMPLETATA — La connessione del giocatore è stata interrotta. • TIMEDOUT: è stata ricevuta una richiesta di sessione, ma il giocatore non si è connesso o non è stato convalidato entro il limite di timeout (60 secondi). Tipo: FString Obbligatorio: no
NextToken	Il token che indica l'inizio della pagina successiva di risultati. Per specificare l'inizio del set di risultati, non fornire un valore. Se fornisci un ID di sessione del giocatore, questo parametro viene ignorato. Tipo: FString Obbligatorio: no
Limite	Numero massimo di risultati da restituire. Se fornisci un ID di sessione del giocatore, questo parametro viene ignorato. Tipo: int Obbligatorio: no

FStopMatchBackfillRequest

Informazioni utilizzate per annullare una richiesta di matchmaking backfill. Il server di gioco comunica queste informazioni al Amazon GameLift Servers servizio durante una chiamata. [StopMatchBackfill\(\)](#)

Properties	Descrizione
GameSessionArn	Un identificatore univoco della sessione di gioco della richiesta annullata. Tipo: FString Obbligatorio: sì
MatchmakingConfigurationArn	Un identificatore univoco del matchmaker a cui è stata inviata questa richiesta. Tipo: FString Obbligatorio: sì
TicketId	Un identificatore univoco del ticket di richiesta di riempimento da annullare. Tipo: FString Obbligatorio: sì

FAttributeValore

Utilizzate questi valori nelle coppie chiave-valore [FPlayer](#) degli attributi. Questo oggetto consente di specificare il valore di un attributo utilizzando uno qualsiasi dei tipi di dati validi: stringa, numero, array di stringhe o mappa di dati. Ogni `FAttributeValue` oggetto può utilizzare solo una delle proprietà disponibili.

Properties	Description
AttrType	Specifica il tipo di valore dell'attributo. Tipo: un valore <code>FAttributeType</code> enum. Obbligatorio: no
S	Rappresenta un valore di attributo di tipo stringa.

Properties	Description
	Tipo: FString Obbligatorio: no
N	Rappresenta un valore di attributo numerico. Tipo: double Obbligatorio: no
SL	Rappresenta una matrice di valori di attributi di tipo stringa. Tipo: TArray<FString> Obbligatorio: no
SDM	Rappresenta un dizionario di chiavi di stringa e valori doppi. Tipo: TMap<FString, double> Obbligatorio: no

FGameLiftGetFleetRoleCredentialsRequest

Questo tipo di dati fornisce credenziali di ruolo che estendono l'accesso limitato alle tue AWS risorse al server di gioco. Per ulteriori informazioni, consultare [Configurare un ruolo di servizio IAM per Amazon GameLift Servers](#).

Properties	Descrizione
RoleArn	L'Amazon Resource Name (ARN) del ruolo di servizio che estende l'accesso limitato alle tue AWS risorse. Tipo: FString Obbligatorio: no

Properties	Descrizione
RoleSessionName	Il nome della sessione che descrive l'uso delle credenziali del ruolo. Tipo: FString Obbligatorio: no

FGameLiftLongOutcome

Questo tipo di dati risulta da un'azione e produce un oggetto con le seguenti proprietà:

Properties	Description
Risultato	Il risultato dell'azione. Tipo: long Obbligatorio: no
ResultWithOwnership	Il risultato dell'azione, espresso come rvalue, in modo che il codice chiamante possa assumere la proprietà dell'oggetto. Tipo: long&& Obbligatorio: no
Completato	Se l'azione ha avuto successo o meno. Tipo: bool Obbligatorio: sì
Errore	L'errore che si è verificato se l'azione non ha avuto successo. Tipo: the section called “FGameLiftError” Obbligatorio: no

FGameLiftStringOutcome

Questo tipo di dati risulta da un'azione e produce un oggetto con le seguenti proprietà:

Properties	Description
Risultato	Il risultato dell'azione. Tipo: FString Obbligatorio: no
ResultWithOwnership	Il risultato dell'azione, espresso come rvalue, in modo che il codice chiamante possa assumere la proprietà dell'oggetto. Tipo: FString&& Obbligatorio: no
Completato	Se l'azione ha avuto successo o meno. Tipo: bool Obbligatorio: sì
Errore	L'errore che si è verificato se l'azione non ha avuto successo. Tipo: the section called “FGameLiftError” Obbligatorio: no

FGameLiftDescribePlayerSessionsOutcome

Questo tipo di dati risulta da un'azione e produce un oggetto con le seguenti proprietà:

Properties	Description
Risultato	Il risultato dell'azione. Tipo: the section called “FGameLiftDescribePlayerSessionsResult”

Properties	Description
	Obbligatorio: no
ResultWithOwnership	Il risultato dell'azione, espresso come rvalue, in modo che il codice chiamante possa assumere la proprietà dell'oggetto. Tipo: <code>FGameLiftDescribePlayerSessionsResult&&</code> Obbligatorio: no
Completato	Se l'azione ha avuto successo o meno. Tipo: <code>bool</code> Obbligatorio: sì
Errore	L'errore che si è verificato se l'azione non ha avuto successo. Tipo: the section called “FGameLiftError” Obbligatorio: no

FGameLiftDescribePlayerSessionsResult

Properties	Description
PlayerSessions	Tipo: <code>TArray<FGameLiftPlayerSession></code> Obbligatorio: sì
NextToken	Il token che indica l'inizio della pagina successiva dei risultati. Per specificare l'inizio del set di risultati, non fornire un valore. Se fornisci un ID di sessione del giocatore, questo parametro viene ignorato. Tipo: <code>FString</code> Obbligatorio: no

Properties	Description
Completato	Se l'azione ha avuto successo o meno. Tipo: bool Obbligatorio: sì
Errore	L'errore che si è verificato se l'azione non ha avuto successo. Tipo: the section called “FGameLiftError” Obbligatorio: no

FGenericRisultato

Questo tipo di dati risulta da un'azione e produce un oggetto con le seguenti proprietà:

Properties	Description
Completato	Se l'azione ha avuto successo o meno. Tipo: bool Obbligatorio: sì
Errore	L'errore che si è verificato se l'azione non ha avuto successo. Tipo: the section called “FGameLiftError” Obbligatorio: no

FGameLiftPlayerSession

Properties	Description
CreationTime	Tipo: long Obbligatorio: sì

Properties	Description
FleetId	Tipo: FString Obbligatorio: sì
GameSessionId	Tipo: FString Obbligatorio: sì
IpAddress	Tipo: FString Obbligatorio: sì
PlayerData	Tipo: FString Obbligatorio: sì
PlayerId	Tipo: FString Obbligatorio: sì
PlayerSessionId	Tipo: FString Obbligatorio: sì
Porta	Tipo: int Obbligatorio: sì
Status	Tipo: Un PlayerSessionStatus enum . Obbligatorio: sì
TerminationTime	Tipo: long Obbligatorio: sì
DnsName	Tipo: FString Obbligatorio: sì

FGameLiftGetComputeCertificateOutcome

Questo tipo di dati risulta da un'azione e produce un oggetto con le seguenti proprietà:

Properties	Description
Risultato	Il risultato dell'azione. Tipo: the section called “FGameLiftGetComputeCertificateResult” Obbligatorio: no
ResultWithOwnership	Il risultato dell'azione, espresso come rvalue, in modo che il codice chiamante possa assumere la proprietà dell'oggetto. Tipo: FGameLiftGetComputeCertificateResult&& Obbligatorio: no
Completato	Se l'azione ha avuto successo o meno. Tipo: bool Obbligatorio: sì
Errore	L'errore che si è verificato se l'azione non ha avuto successo. Tipo: the section called “FGameLiftError” Obbligatorio: no

FGameLiftGetComputeCertificateResult

Il percorso del certificato TLS sul computer e il nome host del calcolo.

Properties	Description
CertificatePath	Tipo: FString Obbligatorio: sì

Properties	Description
ComputeName	Tipo: FString Obbligatorio: sì

FGameLiftGetFleetRoleCredentialsOutcome

Questo tipo di dati risulta da un'azione e produce un oggetto con le seguenti proprietà:

Properties	Description
Risultato	Il risultato dell'azione. Tipo: the section called “FGetFleetRoleCredentialsResult” Obbligatorio: no
ResultWithOwnership	Il risultato dell'azione, espresso come rvalue, in modo che il codice chiamante possa assumere la proprietà dell'oggetto. Tipo: FGameLiftGetFleetRoleCredentialsResult&& Obbligatorio: no
Completato	Se l'azione ha avuto successo o meno. Tipo: bool Obbligatorio: sì
Errore	L'errore che si è verificato se l'azione non ha avuto successo. Tipo: the section called “FGameLiftError” Obbligatorio: no

FGetFleetRoleCredentialsResult

Properties	Description
AccessKeyId	L'ID della chiave di accesso per autenticare e fornire l'accesso alle AWS risorse. Tipo: FString Obbligatorio: no
AssumedRoleId	L'ID dell'utente a cui appartiene il ruolo di servizio. Tipo: FString Obbligatorio: no
AssumedRoleUserArn	L'Amazon Resource Name (ARN) dell'utente a cui appartiene il ruolo di servizio. Tipo: FString Obbligatorio: no
Scadenza	Il periodo di tempo che manca alla scadenza delle credenziali di sessione. Tipo: FDateTime Obbligatorio: no
SecretAccessKey	L'ID della chiave di accesso segreta per l'autenticazione. Tipo: FString Obbligatorio: no
SessionToken	Un token per identificare la sessione attiva corrente che interagisce con AWS le tue risorse. Tipo: FString

Properties	Description
	Obbligatorio: no
Completato	Se l'azione ha avuto successo o meno. Tipo: bool Obbligatorio: sì
Errore	L'errore che si è verificato se l'azione non ha avuto successo. Tipo: the section called “GameLiftError” Obbligatorio: no

FGameLiftError

Properties	Description
ErrorType	Il tipo di errore. Tipo: Un GameLiftErrorType enum . Obbligatorio: no
ErrorMessage	Il nome dell'errore. Tipo: std::string Obbligatorio: no
ErrorMessage	Messaggio di errore. Tipo: std::string Obbligatorio: no

enumerazioni;

Le enumerazioni definite per l'SDK del server per Amazon GameLift Servers (Unreal) sono definite come segue:

FAttributeTipo

- NONE
- STRINGA
- DOPPIA
- ELENCO_STRINGHE
- STRING_DOUBLE_MAP

GameLiftErrorType

Valore di stringa che indica il tipo di errore. I valori validi includono:

- SERVICE_CALL_FAILED — Una chiamata a un servizio non è riuscita. AWS
- LOCAL_CONNECTION_FAILED — La connessione locale a non è riuscita. Amazon GameLift Servers
- NETWORK_NOT_INITIALIZED — La rete non è stata inizializzata.
- GAMESESSION_ID_NOT_SET — L'ID della sessione di gioco non è stato impostato.
- BAD_REQUEST_EXCEPTION
- ECCEZIONE_DI_SERVIZIO INTERNA
- ALREADY_INITIALIZED — Il Amazon GameLift Servers server o il client è già stato inizializzato con Initialize ().
- FLEET_MISMATCH — La flotta bersaglio non corrisponde alla flotta di una GameSession o PlayerSession.
- GAMELIFT_CLIENT_NOT_INITIALIZED — Il client non è stato inizializzato. Amazon GameLift Servers
- GAMELIFT_SERVER_NOT_INITIALIZED — Amazon GameLift Servers Il server non è stato inizializzato.
- GAME_SESSION_ENDED_FAILED: l'SDK del server non è riuscito a contattare il servizio per segnalare la fine della sessione di gioco. Amazon GameLift Servers
- GAME_SESSION_NOT_READY — La sessione di gioco sul server non è stata attivata. Amazon GameLift Servers

- `GAME_SESSION_READY_FAILED`: l'SDK del server per cui non è stato possibile contattare il servizio per segnalare che la sessione di gioco è pronta. Amazon GameLift Servers
- `INITIALIZATION_MISMATCH` — È stato chiamato un metodo client dopo `Server::Initialize()` o viceversa.
- `NOT_INITIALIZED` — Il Amazon GameLift Servers server o il client non sono stati inizializzati con `Initialize()`.
- `NO_TARGET_ALIASID_SET` — Non è stato impostato un `aliasID` di destinazione.
- `NO_TARGET_FLEET_SET` — Non è stata impostata una flotta di obiettivi.
- `PROCESS_ENDING_FAILED`: l'SDK del server non è riuscito a contattare il servizio per segnalare la fine del processo. Amazon GameLift Servers
- `PROCESS_NOT_ACTIVE` — Il processo del server non è ancora attivo, non è associato a un processo e non può accettarlo o elaborarlo. `GameSession PlayerSessions`
- `PROCESS_NOT_READY` — Il processo del server non è ancora pronto per essere attivato.
- `PROCESS_READY_FAILED`: l'SDK del server non è riuscito a contattare il servizio per Amazon GameLift Servers segnalare che il processo è pronto.
- `SDK_VERSION_DETECTION_FAILED` — Rilevamento della versione SDK non riuscito.
- `STX_CALL_FAILED` — Una chiamata al componente di backend del server non è riuscita. `XStx`
- `STX_INITIALIZATION_FAILED` — L'inizializzazione del componente di backend del server non è riuscita. `XStx`
- `UNEXPECTED_PLAYER_SESSION` — Il server ha rilevato una sessione di giocatore non registrata.
- `WEBSOCKET_CONNECT_FAILURE`
- `WEBSOCKET_CONNECT_FAILURE_FORBIDDEN`
- `URL NON VALIDO DI WEBSOCKET_CONNECT_FAILURE_INVALID_`
- `WEBSOCKET_CONNECT_FAILURE_TIMEOUT`
- `WEBSOCKET_RETRIABLE_SEND_MESSAGE_FAILURE` — Errore recuperabile nell'invio di un messaggio al servizio. `GameLift WebSocket`
- `WEBSOCKET_SEND_MESSAGE_FAILURE` — Errore nell'invio di un messaggio al GameLift servizio. `WebSocket`
- `MATCH_BACKFILL_REQUEST_VALIDATION` — La convalida della richiesta non è riuscita.
- `PLAYER_SESSION_REQUEST_VALIDATION` — La convalida della richiesta non è riuscita.

EPlayerSessionCreationPolicy

Valore della stringa che indica se la sessione di gioco accetta nuovi giocatori. I valori validi includono:

- ACCEPT_ALL: accetta tutte le nuove sessioni giocatore.
- DENY_ALL: rifiuta tutte le nuove sessioni giocatore.
- NOT_SET — La sessione di gioco non è impostata per accettare o rifiutare sessioni di nuovi giocatori.

EPlayerSessionStatus

- ACTIVE
- COMPLETED
- NOT_SET
- RISERVATO
- TIMEOUT

[Server C++ \(Unreal\) SDK 5.x per Amazon GameLift Servers -- Tipi di dati](#)

Argomenti

- [GetSdkVersion\(\)](#)
- [InitSDK\(\)](#)
- [InitSDK\(\)](#)
- [ProcessReady\(\)](#)
- [ProcessEnding\(\)](#)
- [ActivateGameSession\(\)](#)
- [UpdatePlayerSessionCreationPolicy\(\)](#)
- [GetGameSessionId\(\)](#)
- [GetTerminationTime\(\)](#)
- [AcceptPlayerSession\(\)](#)
- [RemovePlayerSession\(\)](#)
- [DescribePlayerSessions\(\)](#)
- [StartMatchBackfill\(\)](#)
- [StopMatchBackfill\(\)](#)

- [GetComputeCertificate\(\)](#)
- [GetFleetRoleCredentials\(\)](#)
- [Distruggi \(\)](#)

GetSdkVersion()

Restituisce il numero di versione corrente dell'SDK integrato nel processo del server.

Sintassi

```
FGameLiftStringOutcome GetSdkVersion();
```

Valore restituito

Se l'esito è positivo, restituisce la versione corrente dell'SDK come oggetto [the section called "FGameLiftStringOutcome"](#). L'oggetto restituito include il numero di versione (esempio `5.0.0`). Se l'esito è negativo, verrà restituito un messaggio di errore.

Esempio

```
Aws::GameLift::AwsStringOutcome SdkVersionOutcome =  
    Aws::GameLift::Server::GetSdkVersion();
```

InitSDK()

Inizializza l'Amazon GameLift Servers SDK per una flotta EC2 gestita. Richiama questo metodo all'avvio, prima che si verifichi qualsiasi altra inizializzazione correlata a Amazon GameLift Servers. Questo metodo legge i parametri del server dall'ambiente host per configurare la comunicazione tra il server e il Amazon GameLift Servers servizio. Utilizza un token di idempotenza, quindi puoi riprovare a eseguire questa chiamata in tutta sicurezza quando fallisce.

Sintassi

```
FGameLiftGenericOutcome InitSDK()
```

Valore restituito

In caso di successo, restituisce un `InitSdkOutcome` oggetto che indica che il processo del server è pronto per la chiamata. [ProcessReady\(\)](#)

Esempio

```
//Call InitSDK to establish a local connection with the Amazon GameLift Servers Agent  
to enable further communication.  
FGameLiftGenericOutcome initSdkOutcome = GameLiftSdkModule->InitSDK();
```

InitSDK()

Inizializza l'Amazon GameLift ServersSDK per una flotta Anywhere o una flotta di container gestita. Richiama questo metodo all'avvio, prima che si verifichi qualsiasi altra inizializzazione correlata a. Amazon GameLift Servers Questo metodo richiede parametri espliciti del server per configurare la comunicazione tra il server e il Amazon GameLift Servers servizio. Utilizza un token di idempotenza, quindi puoi riprovare a ripetere questa chiamata in tutta sicurezza quando fallisce.

Sintassi

```
FGameLiftGenericOutcome InitSDK(serverParameters)
```

Parameters

[FServerParametri](#)

Per inizializzare un server di gioco su una flotta Amazon GameLift Servers Anywhere, costruisci un `ServerParameters` oggetto con le seguenti informazioni:

- L'URL WebSocket utilizzato per connettersi al server di gioco.
- L'ID del processo utilizzato per ospitare il server di gioco.
- L'ID del computer che ospita i processi del server di gioco.
- L'ID della Amazon GameLift Servers flotta contenente il tuo computer Amazon GameLift Servers Anywhere.
- Il token di autorizzazione generato dall'Amazon GameLift Serversoperazione.

Valore restituito

In caso di successo, restituisce un `InitSdkOutcome` oggetto che indica che il processo del server è pronto per la chiamata [ProcessReady\(\)](#).

Note

Se le chiamate a non `InitSDK()` riescono per le build di gioco distribuite sulle flotte Anywhere, controlla il `ServerSdkVersion` parametro utilizzato durante la creazione della risorsa di compilazione. È necessario impostare esplicitamente questo valore sulla versione SDK del server in uso. Il valore predefinito per questo parametro è 4.x, che non è compatibile. Per risolvere questo problema, crea una nuova build e distribuiscila in una nuova flotta.

Esempio

```
//Define the server parameters
FServerParameters serverParameters;
parameters.m_authToken = "1111aaaa-22bb-33cc-44dd-5555eeee66ff";
parameters.m_fleetId = "arn:aws:gamelift:us-west-1:111122223333:fleet/
fleet-9999ffff-88ee-77dd-66cc-5555bbbb44aa";
parameters.m_hostId = "HardwareAnywhere";
parameters.m_processId = "PID1234";
parameters.m_webSocketUrl = "wss://us-west-1.api.amazongamelift.com";

//Call InitSDK to establish a local connection with the Amazon GameLift Servers Agent
to enable further communication.
FGameLiftGenericOutcome initSdkOutcome = GameLiftSdkModule->InitSDK(serverParameters);
```

ProcessReady()

Notifica Amazon GameLift Servers che il processo del server è pronto per ospitare sessioni di gioco. Chiama questo metodo dopo averlo [InitSDK\(\)](#) invocato. Questo metodo deve essere chiamato solo una volta per processo.

Sintassi

```
GenericOutcome ProcessReady(const Aws::GameLift::Server::ProcessParameters
&processParameters);
```

Parameters**processParameters**

Un [FProcessParametri](#) oggetto che comunica le seguenti informazioni sul processo del server:

- Nomi dei metodi di callback implementati nel codice del server di gioco richiamato dal Amazon GameLift Servers servizio per comunicare con il processo del server.
- Numero di porta sulla quale è in ascolto il processo del server.
- Percorso verso qualsiasi file specifico di una sessione di gioco che Amazon GameLift Servers deve acquisire e archiviare.

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

Questo esempio illustra la chiamata [ProcessReady\(\)](#) e le implementazioni della funzione delegata.

```
//Calling ProcessReady tells Amazon GameLift Servers this game server is ready to
receive incoming game sessions!
UE_LOG(GameServerLog, Log, TEXT("Calling Process Ready"));
FGameLiftGenericOutcome processReadyOutcome = GameLiftSdkModule-
>ProcessReady(*params);
```

ProcessEnding()

Notifica Amazon GameLift Servers che il processo del server sta terminando. Chiama questo metodo dopo tutte le altre attività di pulizia (inclusa la chiusura della sessione di gioco attiva) e prima di terminare il processo. A seconda del risultato di `ProcessEnding()`, il processo termina con successo (0) o errore (-1) e genera un evento relativo alla flotta. Se il processo termina con un errore, l'evento di flotta generato è `SERVER_PROCESS_TERMINATED_UNHEALTHY`.

Sintassi

```
FGameLiftGenericOutcome ProcessEnding()
```

Valore restituito

Restituisce un risultato generico costituito da successo o fallimento con un messaggio di errore.

Esempio

```
//OnProcessTerminate callback. Amazon GameLift Servers will invoke this callback before
shutting down an instance hosting this game server.
```

```
//It gives this game server a chance to save its state, communicate with services,  
etc., before being shut down.  
//In this case, we simply tell Amazon GameLift Servers we are indeed going to shutdown.  
params->OnTerminate.BindLambda(=[]() {  
    UE_LOG(GameServerLog, Log, TEXT("Game Server Process is terminating"));  
    GameLiftSdkModule->ProcessEnding();  
});
```

ActivateGameSession()

Notifica Amazon GameLift Servers che il processo del server ha attivato una sessione di gioco ed è ora pronto a ricevere le connessioni dei giocatori. Questa azione dovrebbe essere richiamata come parte della funzione di `onStartGameSession()` callback, dopo l'inizializzazione di tutta la sessione di gioco.

Sintassi

```
FGameLiftGenericOutcome ActivateGameSession()
```

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

Questo esempio mostra le `ActivateGameSession()` chiamate come parte della funzione `onStartGameSession()` delegata.

```
//When a game session is created, Amazon GameLift Servers sends an activation request  
to the game server and passes along the game session object containing game properties  
and other settings.  
//Here is where a game server should take action based on the game session object.  
//Once the game server is ready to receive incoming player connections, it should  
invoke GameLiftServerAPI.ActivateGameSession()  
auto onGameSession = [](Aws::GameLift::Server::Model::GameSession gameSession)  
{  
    FString gameSessionId = FString(gameSession.GetGameSessionId());  
    UE_LOG(GameServerLog, Log, TEXT("GameSession Initializing: %s"), *gameSessionId);  
    GameLiftSdkModule->ActivateGameSession();  
};
```


UpdatePlayerSessionCreationPolicy()

Aggiorna la capacità della sessione di gioco corrente di accettare nuove sessioni giocatore. Una sessione di gioco può essere configurata per accettare o rifiutare tutte le nuove sessioni giocatore.

Sintassi

```
FGameLiftGenericOutcome UpdatePlayerSessionCreationPolicy(EPlayerSessionCreationPolicy policy)
```

Parameters

playerCreationSessionPolitica

Valore della stringa che indica se la sessione di gioco accetta nuovi giocatori.

I valori validi includono:

- **ACCEPT_ALL**: accetta tutte le nuove sessioni giocatore.
- **DENY_ALL**: rifiuta tutte le nuove sessioni giocatore.

Valore restituito

Restituisce un risultato generico costituito da successo o fallimento con un messaggio di errore.

Esempio

Questo esempio definisce la policy di partecipazione alla sessione di gioco corrente per accettare tutti i giocatori.

```
FGameLiftGenericOutcome outcome = GameLiftSdkModule->UpdatePlayerSessionCreationPolicy(Aws::GameLift::Model::EPlayerSessionCreationPolicy::ACCEPT_A
```

GetGameSessionId()

Recupera l'ID della sessione di gioco ospitata dal processo del server attivo.

Per i processi inattivi che non vengono attivati con una sessione di gioco, la chiamata restituisce un [the section called “FGameLiftError”](#)

Sintassi

```
FGameLiftStringOutcome GetGameSessionId()
```

Parameters

Questa operazione non prevede parametri.

Valore restituito

Se l'esito è positivo, l'ID della sessione di gioco verrà restituito come oggetto [the section called "FGameLiftStringOutcome"](#). Se non riesce, restituisce un messaggio di errore».

Per i processi inattivi che non vengono attivati con una sessione di gioco, la chiamata restituisce `Success = True` e `GameSessionId = ""`.

Esempio

```
//When a game session is created, Amazon GameLift Servers sends an activation request
//to the game server and passes along the game session object containing game properties
//and other settings.
//Here is where a game server should take action based on the game session object.
//Once the game server is ready to receive incoming player connections, it should
//invoke GameLiftServerAPI.ActivateGameSession()
auto onGameSession = [](Aws::GameLift::Server::Model::GameSession gameSession)
{
    FString gameId = FString(gameSession.GetGameSessionId());
    UE_LOG(GameServerLog, Log, TEXT("GameSession Initializing: %s"), *gameId);
    GameLiftSdkModule->ActivateGameSession();
};
```

GetTerminationTime()

Restituisce il tempo di arresto pianificato di un processo del server, se è disponibile un tempo di chiusura. Un processo server interviene dopo aver ricevuto una `onProcessTerminate()` richiamata da Amazon GameLift Servers. Amazon GameLift Servers chiama `onProcessTerminate()` per i seguenti motivi:

- Quando il processo del server ha segnalato problemi di salute o non ha risposto. Amazon GameLift Servers
- Quando si termina l'istanza durante un evento di scale-down.

- [Quando un'istanza viene terminata a causa di un'interruzione di un'istanza spot.](#)

Sintassi

```
AwsDateTimeOutcome GetTerminationTime()
```

Valore restituito

In caso di successo, restituisce l'ora di terminazione come oggetto. `AwsDateTimeOutcome` Il valore è il tempo di terminazione, espresso in segni di spunta trascorsi da allora. `0001 00:00:00` Ad esempio, il valore della data e dell'ora è uguale ai segni di `2020-09-13 12:26:40 -000Z` spunta. `6373559680000000000` Se non è disponibile alcun orario di terminazione, restituisce un messaggio di errore.

Se il processo non ha ricevuto una `ProcessParameters.OnProcessTerminate()` richiamata, viene restituito un messaggio di errore. Per ulteriori informazioni sulla chiusura di un processo server, vedere. [Rispondi a una notifica di chiusura del processo del server](#)

Esempio

```
AwsDateTimeOutcome TermTimeOutcome = GameLiftSdkModule->GetTerminationTime();
```

AcceptPlayerSession()

Notifica Amazon GameLift Servers che un giocatore con l'ID di sessione specificato si è connesso al processo del server e deve essere convalidato. Amazon GameLift Servers verifica che l'ID della sessione del giocatore sia valido. Dopo la convalida della sessione di gioco, Amazon GameLift Servers modifica lo stato dello slot del giocatore da RISERVATO ad ATTIVO.

Sintassi

```
FGameLiftGenericOutcome AcceptPlayerSession(const FString& playerId)
```

Parameters

playerSessionId

ID univoco rilasciato Amazon GameLift Servers quando viene creata una nuova sessione giocatore.

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

Questo esempio gestisce una richiesta di connessione che include la convalida e il rifiuto di una sessione giocatore non valida. IDs

```
bool GameLiftManager::AcceptPlayerSession(const FString& playerSessionId, const
    FString& playerId)
{
    #if WITH_GAMELIFT
        UE_LOG(GameServerLog, Log, TEXT("Accepting GameLift PlayerSession: %s . PlayerId:
%s"), *playerSessionId, *playerId);
        FString gsId = GetCurrentGameSessionId();
        if (gsId.IsEmpty()) {
            UE_LOG(GameServerLog, Log, TEXT("No GameLift GameSessionId. Returning early!"));
            return false;
        }

        if (!GameLiftSdkModule->AcceptPlayerSession(playerSessionId).IsSuccess()) {
            UE_LOG(GameServerLog, Log, TEXT("PlayerSession not Accepted.));
            return false;
        }

        // Add PlayerSession from internal data structures keeping track of connected players
        connectedPlayerSessionIds.Add(playerSessionId);
        idToPlayerSessionMap.Add(playerSessionId, PlayerSession{ playerId,
playerSessionId });
        return true;
    #else
        return false;
    #endif
}
```

RemovePlayerSession()

Notifica Amazon GameLift Servers che un giocatore si è disconnesso dal processo del server. In risposta, Amazon GameLift Servers modifica lo slot del giocatore rendendolo disponibile.

Sintassi

```
FGameLiftGenericOutcome RemovePlayerSession(const FString& playerId)
```

Parameters

playerSessionId

ID univoco rilasciato Amazon GameLift Servers quando viene creata una nuova sessione di gioco.

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

```
bool GameLiftManager::RemovePlayerSession(const FString& playerId)
{
    #if WITH_GAMELIFT
        UE_LOG(GameServerLog, Log, TEXT("Removing GameLift PlayerSession: %s"),
            *playerSessionId);

        if (!GameLiftSdkModule->RemovePlayerSession(playerSessionId).IsSuccess()) {
            UE_LOG(GameServerLog, Log, TEXT("PlayerSession Removal Failed"));
            return false;
        }

        // Remove PlayerSession from internal data structures that are keeping track of
        connected players
        connectedPlayerSessionIds.Remove(playerSessionId);
        idToPlayerSessionMap.Remove(playerSessionId);

        // end the session if there are no more players connected
        if (connectedPlayerSessionIds.Num() == 0) {
            EndSession();
        }

        return true;
    #else
        return false;
    #endif
}
```

DescribePlayerSessions()

Recupera i dati della sessione del giocatore che includono impostazioni, metadati della sessione e dati del giocatore. Utilizzate questo metodo per ottenere informazioni su quanto segue:

- Una sessione per giocatore singolo
- Tutte le sessioni dei giocatori in una sessione di gioco
- Tutte le sessioni dei giocatori sono associate a un ID giocatore singolo

Sintassi

```
FGameLiftDescribePlayerSessionsOutcome DescribePlayerSessions(const  
FGameLiftDescribePlayerSessionsRequest &describePlayerSessionsRequest)
```

Parameters

[FGameLiftDescribePlayerSessionsRequest](#)

Un [the section called “FGameLiftDescribePlayerSessionsRequest”](#) oggetto che descrive le sessioni dei giocatori da recuperare.

Valore restituito

Se l'esito è positivo, restituisce un oggetto [the section called “FGameLiftDescribePlayerSessionsOutcome”](#) contenente un set di oggetti di sessione giocatore corrispondente ai parametri della richiesta.

Esempio

Questo esempio richiede che tutte le sessioni dei giocatori siano connesse attivamente a una sessione di gioco specificata. Omettendo NextToken e impostando il valore Limite su 10, Amazon GameLift Servers restituisce i primi 10 record di sessione tra giocatori corrispondenti alla richiesta.

```
void GameLiftManager::DescribePlayerSessions()  
{  
    #if WITH_GAMELIFT  
    FString localPlayerSessions;  
    for (auto& psId : connectedPlayerSessionIds)  
    {  
        PlayerSession ps = idToPlayerSessionMap[psId];  
    }  
}
```

```

    localPlayerSessions += FString::Printf(TEXT("%s : %s  ; "), *(ps.playerSessionId),
*(ps.playerId));
}
UE_LOG(GameServerLog, Log, TEXT("LocalPlayerSessions: %s"), *localPlayerSessions);

UE_LOG(GameServerLog, Log, TEXT("Describing PlayerSessions in this GameSession"));
FGameLiftDescribePlayerSessionsRequest request;
request.m_gameSessionId = GetCurrentGameSessionId();

FGameLiftDescribePlayerSessionsOutcome outcome = GameLiftSdkModule-
>DescribePlayerSessions(request);
LogDescribePlayerSessionsOutcome(outcome);
#endif
}

```

StartMatchBackfill()

Invia una richiesta per trovare nuovi giocatori per gli slot aperti in una sessione di gioco creata con FlexMatch. Per ulteriori informazioni, consulta la funzione di [FlexMatchriempimento](#).

Questa operazione è asincrona. Se vengono abbinati nuovi giocatori, Amazon GameLift Servers fornisce dati aggiornati sul matchmaker utilizzando la funzione di callback.

OnUpdateGameSession()

Un processo del server può avere un solo backfill degli abbinamenti attivo alla volta. Per inviare una nuova richiesta, chiama prima [StopMatchBackfill\(\)](#) per annullare la richiesta originale.

Sintassi

```

FGameLiftStringOutcome StartMatchBackfill (FStartMatchBackfillRequest
&startBackfillRequest);

```

Parameters

[FStartMatchBackfillRequest](#)

Un StartMatchBackfillRequest oggetto che comunica le seguenti informazioni:

- ID del ticket da assegnare alla richiesta di backfill. Queste informazioni sono facoltative; se non viene fornito alcun ID, ne Amazon GameLift Servers genererà uno.
- Matchmaker a cui inviare la richiesta. L'ARN di configurazione completo è obbligatorio. Questo valore si trova nei dati del matchmaker della sessione di gioco.

- L'ID della sessione di gioco da riempire.
- I dati di matchmaking disponibili per i giocatori attuali della sessione di gioco.

Valore restituito

Restituisce un `StartMatchBackfillOutcome` oggetto con l'ID del ticket match backfill, oppure restituisce un errore con un messaggio di errore.

Esempio

```
FGameLiftStringOutcome FGameLiftServerSDKModule::StartMatchBackfill(const
    FStartMatchBackfillRequest& request)
{
    #if WITH_GAMELIFT
    Aws::GameLift::Server::Model::StartMatchBackfillRequest sdkRequest;
    sdkRequest.SetTicketId(TCHAR_TO_UTF8(*request.m_ticketId));
    sdkRequest.SetGameSessionArn(TCHAR_TO_UTF8(*request.m_gameSessionArn));

    sdkRequest.SetMatchmakingConfigurationArn(TCHAR_TO_UTF8(*request.m_matchmakingConfigurationArn));
    for (auto player : request.m_players) {
        Aws::GameLift::Server::Model::Player sdkPlayer;
        sdkPlayer.SetPlayerId(TCHAR_TO_UTF8(*player.m_playerId));
        sdkPlayer.SetTeam(TCHAR_TO_UTF8(*player.m_team));
        for (auto entry : player.m_latencyInMs) {
            sdkPlayer.WithLatencyMs(TCHAR_TO_UTF8(*entry.Key), entry.Value);
        }

        std::map<std::string, Aws::GameLift::Server::Model::AttributeValue>
        sdkAttributeMap;
        for (auto attributeEntry : player.m_playerAttributes) {
            FAttributeValue value = attributeEntry.Value;
            Aws::GameLift::Server::Model::AttributeValue attribute;
            switch (value.m_type) {
                case FAttributeType::STRING:
                    attribute =
            Aws::GameLift::Server::Model::AttributeValue(TCHAR_TO_UTF8(*value.m_S));
                    break;
                case FAttributeType::DOUBLE:
                    attribute = Aws::GameLift::Server::Model::AttributeValue(value.m_N);
                    break;
                case FAttributeType::STRING_LIST:
```



```

        attribute =
        Aws::GameLift::Server::Model::AttributeValue::ConstructStringList();
        for (auto sl : value.m_SL) {
            attribute.AddString(TCHAR_TO_UTF8(*sl));
        };
        break;
        case FAttributeType::STRING_DOUBLE_MAP:
            attribute =
            Aws::GameLift::Server::Model::AttributeValue::ConstructStringDoubleMap();
            for (auto sdm : value.m_SDM) {
                attribute.AddStringAndDouble(TCHAR_TO_UTF8(*sdm.Key), sdm.Value);
            };
            break;
        }
        sdkPlayer.WithPlayerAttribute((TCHAR_TO_UTF8(*attributeEntry.Key)), attribute);
    }
    sdkRequest.AddPlayer(sdkPlayer);
}
auto outcome = Aws::GameLift::Server::StartMatchBackfill(sdkRequest);
if (outcome.IsSuccess()) {
    return FGameLiftStringOutcome(outcome.GetResult().GetTicketId());
}
else {
    return FGameLiftStringOutcome(FGameLiftError(outcome.GetError()));
}
#else
return FGameLiftStringOutcome("");
#endif
}

```

StopMatchBackfill()

Annulla una richiesta di match backfill attiva. Per ulteriori informazioni, consulta la funzione di [FlexMatchriempimento](#).

Sintassi

```

FGameLiftGenericOutcome StopMatchBackfill (FStopMatchBackfillRequest
&stopBackfillRequest);

```

Parameters

FStopMatchBackfillRequest

Un StopMatchBackfillRequest oggetto che identifica il ticket di matchmaking da annullare:

- L'ID del ticket assegnato alla richiesta di riempimento.
- Il matchmaker a cui è stata inviata la richiesta di riempimento.
- La sessione di gioco associata alla richiesta di riempimento.

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

```
FGameLiftGenericOutcome FGameLiftServerSDKModule::StopMatchBackfill(const
FStopMatchBackfillRequest& request)
{
    #if WITH_GAMELIFT
    Aws::GameLift::Server::Model::StopMatchBackfillRequest sdkRequest;
    sdkRequest.SetTicketId(TCHAR_TO_UTF8(*request.m_ticketId));
    sdkRequest.SetGameSessionArn(TCHAR_TO_UTF8(*request.m_gameSessionArn));

    sdkRequest.SetMatchmakingConfigurationArn(TCHAR_TO_UTF8(*request.m_matchmakingConfigurationArn));
    auto outcome = Aws::GameLift::Server::StopMatchBackfill(sdkRequest);
    if (outcome.IsSuccess()) {
        return FGameLiftGenericOutcome(nullptr);
    }
    else {
        return FGameLiftGenericOutcome(FGameLiftError(outcome.GetError()));
    }
    #else
    return FGameLiftGenericOutcome(nullptr);
    #endif
}
```

GetComputeCertificate()

Recupera il percorso del certificato TLS utilizzato per crittografare la connessione di rete tra la risorsa di elaborazione Amazon GameLift Servers Anywhere e. Amazon GameLift Servers Puoi utilizzare

il percorso del certificato quando registri il tuo dispositivo di elaborazione in una flotta Anywhere. Amazon GameLift Servers Per ulteriori informazioni, consultare [RegisterCompute](#).

Sintassi

```
FGameLiftGetComputeCertificateOutcome FGameLiftServerSDKModule::GetComputeCertificate()
```

Valore restituito

Restituisce un `GetComputeCertificateResponse` oggetto contenente quanto segue:

- `CertificatePath`: il percorso del certificato TLS sulla risorsa di calcolo.
- `HostName`: il nome host della risorsa di elaborazione.

Esempio

```
FGameLiftGetComputeCertificateOutcome FGameLiftServerSDKModule::GetComputeCertificate()
{
    #if WITH_GAMELIFT
    auto outcome = Aws::GameLift::Server::GetComputeCertificate();
    if (outcome.IsSuccess()) {
        auto& outres = outcome.GetResult();
        FGameLiftGetComputeCertificateResult result;
        result.m_certificate_path = UTF8_TO_TCHAR(outres.GetCertificatePath());
        result.m_computeName = UTF8_TO_TCHAR(outres.GetComputeName());
        return FGameLiftGetComputeCertificateOutcome(result);
    }
    else {
        return FGameLiftGetComputeCertificateOutcome(FGameLiftError(outcome.GetError()));
    }
    #else
    return FGameLiftGetComputeCertificateOutcome(FGameLiftGetComputeCertificateResult());
    #endif
}
```

GetFleetRoleCredentials()

Recupera le credenziali del ruolo IAM che autorizzano Amazon GameLift Servers a interagire con altri. Servizi AWS Per ulteriori informazioni, consulta [Connect il server di gioco Amazon GameLift Servers ospitato ad altre AWS risorse](#).

Sintassi

```
FGameLiftGetFleetRoleCredentialsOutcome
FGameLiftServerSDKModule::GetFleetRoleCredentials(const
FGameLiftGetFleetRoleCredentialsRequest &request)
```

Parameters

[FGameLiftGetFleetRoleCredentialsRequest](#)

Valore restituito

Restituisce un oggetto [the section called “FGameLiftGetFleetRoleCredentialsOutcome”](#).

Esempio

```
FGameLiftGetFleetRoleCredentialsOutcome
FGameLiftServerSDKModule::GetFleetRoleCredentials(const
FGameLiftGetFleetRoleCredentialsRequest &request)
{
    #if WITH_GAMELIFT
    Aws::GameLift::Server::Model::GetFleetRoleCredentialsRequest sdkRequest;
    sdkRequest.SetRoleArn(TCHAR_TO_UTF8(*request.m_roleArn));
    sdkRequest.SetRoleSessionName(TCHAR_TO_UTF8(*request.m_roleSessionName));

    auto outcome = Aws::GameLift::Server::GetFleetRoleCredentials(sdkRequest);

    if (outcome.IsSuccess()) {
        auto& outres = outcome.GetResult();
        FGameLiftGetFleetRoleCredentialsResult result;
        result.m_assumedUserRoleArn = UTF8_TO_TCHAR(outres.GetAssumedUserRoleArn());
        result.m_assumedRoleId = UTF8_TO_TCHAR(outres.GetAssumedRoleId());
        result.m_accessKeyId = UTF8_TO_TCHAR(outres.GetAccessKeyId());
        result.m_secretAccessKey = UTF8_TO_TCHAR(outres.GetSecretAccessKey());
        result.m_sessionToken = UTF8_TO_TCHAR(outres.GetSessionToken());
        result.m_expiration = FDateTime::FromUnixTimestamp(outres.GetExpiration());
        return FGameLiftGetFleetRoleCredentialsOutcome(result);
    }
    else {
        return FGameLiftGetFleetRoleCredentialsOutcome(FGameLiftError(outcome.GetError()));
    }
    #else
    return
    FGameLiftGetFleetRoleCredentialsOutcome(FGameLiftGetFleetRoleCredentialsResult());
```

```
#endif  
}
```

Distruggi ()

Libera l'SDK del server di Amazon GameLift Servers gioco dalla memoria. È consigliabile chiamare questo metodo dopo `ProcessEnding()` e prima di terminare il processo. Se utilizzi una flotta Anywhere e non interrompi i processi del server dopo ogni sessione di gioco, chiama `Destroy()` e poi `InitSDK()` reinizializza prima di notificare Amazon GameLift Servers che il processo è pronto per ospitare una sessione di gioco. `ProcessReady()`

Sintassi

```
FGameLiftGenericOutcome Destroy();
```

Parameters

Non ci sono parametri.

Valore restituito

Restituisce un risultato generico costituito da successo o fallimento con un messaggio di errore.

Esempio

```
// First call ProcessEnding()  
FGameLiftGenericOutcome processEndingOutcome = GameLiftSdkModule->ProcessEnding();  
  
// Then call Destroy() to free the SDK from memory  
FGameLiftGenericOutcome destroyOutcome = GameLiftSdkModule->Destroy();  
  
// Exit the process with success or failure  
if (processEndingOutcome.IsSuccess() && destroyOutcome.IsSuccess())  
{  
    UE_LOG(GameServerLog, Log, TEXT("Server process ending successfully"));  
}  
else {  
    if (!processEndingOutcome.IsSuccess()) {  
        const FGameLiftError& error = processEndingOutcome.GetError();  
        UE_LOG(GameServerLog, Error, TEXT("ProcessEnding() failed. Error: %s"),  
            error.m_errorMessage.IsEmpty() ? TEXT("Unknown error") :  
            *error.m_errorMessage);  
    }  
}
```

```
    }  
    if (!destroyOutcome.IsSuccess()) {  
        const FGameLiftError& error = destroyOutcome.GetError();  
        UE_LOG(GameServerLog, Error, TEXT("Destroy() failed. Error: %s"),  
            error.m_errorMessage.IsEmpty() ? TEXT("Unknown error") :  
            *error.m_errorMessage);  
    }  
}
```

Server SDK per la Amazon GameLift Servers versione 4 e precedenti

Questo riferimento documenta l'SDK del server per la Amazon GameLift Servers versione 4.x e precedenti. L'SDK del server fornisce le funzionalità di base utilizzate dai server di gioco per comunicare con il servizio. Amazon GameLift Servers Ad esempio, il server di gioco riceve dal servizio richieste di avvio e interruzione delle sessioni di gioco e fornisce aggiornamenti regolari sullo stato delle sessioni di gioco al servizio. Integra i tuoi server di gioco con l'SDK del server prima di distribuirli per l'hosting.

Usa questo riferimento all'SDK del server per integrare i tuoi server di gioco multiplayer personalizzati con cui ospitare. Amazon GameLift Servers Per indicazioni sul processo di integrazione, consulta [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#).

Per la versione SDK 4.0.2, puoi scaricarla dalle versioni [ufficiali GitHub](#). Il pacchetto SDK completo che GameLiftLocal include.jar è disponibile nell'artefatto -4.0.2.zip. GameLift-CSharp-ServerSDK

L'ultima versione principale del server SDK for è la 5.x. Amazon GameLift Servers Le seguenti funzionalità di hosting richiedono aggiornamenti alla versione 5.x:

- Amazon GameLift ServersOvunque
- Amazon GameLift Serverscontenitori gestiti
- Amazon GameLift Serversplugin per Unreal Engine e Unity

Argomenti

- [SDK per server C++ per Amazon GameLift Servers 4.x - Azioni](#)
- [SDK del server C# per Amazon GameLift Servers 4.x -- Azioni](#)
- [Server SDK \(Unreal\) per Amazon GameLift Servers -- Azioni](#)

SDK per server C++ per Amazon GameLift Servers 4.x - Azioni

Usa il riferimento all'SDK del server per integrare il tuo gioco multiplayer con cui ospitarlo. Amazon GameLift Servers Per indicazioni sul processo di integrazione, consulta [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#).

Note

Questo riferimento si riferisce a una versione precedente del server SDK for Amazon GameLift Servers. Per la versione più recente, consulta [Server C++ SDK 5.x per -- Azioni Amazon GameLift Servers](#).

SDK per server C++ per Amazon GameLift Servers 4.x -- Tipi di dati

Usa il riferimento all'SDK del server per integrare il tuo gioco multiplayer con cui ospitarlo. Amazon GameLift Servers Per indicazioni sul processo di integrazione, consulta [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#).

Note

Questo riferimento si riferisce a una versione precedente del server SDK for Amazon GameLift Servers. Per la versione più recente, consulta [Server C++ SDK 5.x per Amazon GameLift Servers -- Tipi di dati](#).

L'API è definita in `GameLiftServerAPI.h`, `LogParameters.h` e `ProcessParameters.h`.

[SDK per server C++ per Amazon GameLift Servers 4.x - Azioni](#)

`DescribePlayerSessionsRequest`

Questo tipo di dati viene utilizzato per specificare quale sessione del giocatore recuperare. Puoi utilizzarlo come segue:

- Fornisci un `PlayerSessionId` messaggio per richiedere una sessione di gioco specifica.
- Fornisci un `GameSessionId` campo per richiedere tutte le sessioni dei giocatori nella sessione di gioco specificata.
- Fornisci un `PlayerId` campo per richiedere tutte le sessioni di gioco per il giocatore specificato.

Per le raccolte di grandi dimensioni delle sessioni giocatore, utilizza i parametri di paginazione per recuperare i risultati in blocchi sequenziali.

Indice

GameSessionId

Identificatore univoco della sessione di gioco. Utilizzare questo parametro per richiedere tutte le sessioni giocatore per la sessione di gioco specificata. Il formato dell'ID della sessione di gioco è il seguente: `arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string>`. Il valore di `<ID string>` corrisponde a una stringa ID personalizzata o (se ne è stata specificata una al momento della creazione della sessione di gioco) a una stringa generata.

▪Tipo: stringa

Obbligatorio: no

Limite

Numero massimo di risultati da restituire. Utilizzate questo parametro con `NextToken` per ottenere risultati sotto forma di set di pagine sequenziali. Se viene specificato un ID sessione giocatore, questo parametro verrà ignorato.

Tipo: Integer

Obbligatorio: no

NextToken

Token che indica l'inizio della pagina sequenziale successiva relativa ai risultati. Utilizzare il token restituito con una chiamata precedente a questa operazione. Per specificare l'inizio del set di risultati, non specificare un valore. Se viene specificato un ID sessione giocatore, questo parametro verrà ignorato.

▪Tipo: stringa

Obbligatorio: no

PlayerId

Identificatore univoco per un giocatore. IDs I giocatori sono definiti dallo sviluppatore. Per informazioni, consulta [Genera giocatore IDs](#).

▪Tipo: stringa

Obbligatorio: no

PlayerSessionId

Identificatore univoco della sessione giocatore.

─Tipo: stringa

Obbligatorio: no

PlayerSessionStatusFilter

Stato sessione giocatore su cui filtrare i risultati. Tra gli stati sessione giocatore possibili sono inclusi i seguenti:

- RISERVATO — La richiesta di sessione del giocatore è stata ricevuta, ma il giocatore non si è ancora connesso al server. Il processo and/or è stato convalidato.
- ACTIVE (ATTIVO) - Il giocatore è stato convalidato dal processo del server ed è attualmente collegato.
- COMPLETED (COMPLETATO) - La connessione del giocatore è stata interrotta.
- TIMEDOUT — È stata ricevuta una richiesta di sessione di gioco, ma il giocatore che non si and/or è connesso non è stata convalidata entro il limite di timeout (60 secondi).

─Tipo: stringa

Obbligatorio: no

LogParameters

Questo tipo di dati viene utilizzato per identificare i file generati durante una sessione di gioco che Amazon GameLift Servers caricherà e memorizzerà al termine di questa fase. Queste informazioni vengono comunicate al servizio Amazon GameLift Servers in una chiamata [ProcessReady\(\)](#).

Indice

logPaths

I percorsi di directory ai file di log del server di gioco che desideri che Amazon GameLift Servers archivi per l'accesso futuro. Questi file vengono generati durante ciascuna sessione di gioco. I percorsi e i nomi di file vengono definiti nel server di gioco e archiviati nella directory root della build di gioco. I percorsi di registro devono essere assoluti. Ad esempio, se la build di gioco archivia i log della sessione di gioco in un percorso come `MyGame\sessionlogs\`, il percorso

di log è `c:\game\MyGame\sessionLogs` (su un'istanza Windows) o `/local/game/MyGame/sessionLogs` (su un'istanza Linux).

Tipo: `std::vector<std::string>`

Obbligatorio: no

ProcessParameters

Questo tipo di dati contiene il set di parametri inviato al servizio Amazon GameLift Servers in una chiamata [ProcessReady\(\)](#).

Indice

port

Numero di porta su cui il processo del server rimane in attesa di nuove connessioni dei giocatori. Il valore deve rientrare nella gamma di porte configurate per qualsiasi parco istanze che distribuisce questa build del server di gioco. Questo numero di porta è incluso nella sessione di gioco e negli oggetti della sessione del giocatore utilizzati dalle sessioni di gioco per la connessione a un processo server.

Tipo: numero intero

Obbligatorio: sì

logParameters

Un oggetto con un elenco di percorsi delle directory per i file log delle sessioni di gioco.

Tipo: `Aws::GameLift::Server::` [LogParameters](#)

Obbligatorio: no

onStartGameSessione

Nome della funzione di callback richiamata dal Amazon GameLift Servers servizio per attivare una nuova sessione di gioco. Amazon GameLift Servers chiama questa funzione in risposta alla richiesta del client. [CreateGameSession](#) La funzione di callback passa un [GameSession](#) oggetto (definito nel Amazon GameLift ServersService API Reference).

Tipo: `const std::function<void(Aws::GameLift::Model::GameSession)>`
`onStartGameSession`

Obbligatorio: sì

onProcessTerminate

Nome della funzione di callback richiamata dal Amazon GameLift Servers servizio per forzare l'arresto del processo server. Dopo aver richiamato questa funzione, Amazon GameLift Servers attende per cinque minuti l'arresto del processo del server e risponde con una chiamata [ProcessEnding\(\)](#). Se non viene ricevuta nessuna risposta, arresta il processo del server.

Tipo: `std::function<void()> onProcessTerminate`

Obbligatorio: no

onHealthCheck

Nome della funzione di callback richiamata dal Amazon GameLift Servers servizio per richiedere un rapporto sullo stato di salute del processo del server. Amazon GameLift Servers chiama questa funzione ogni 60 secondi. Dopo aver richiamato questa funzione, Amazon GameLift Servers attende per 60 secondi l'arrivo di una risposta e, se non ne riceve alcuna, registra il processo del server come non integro.

Tipo: `std::function<bool()> onHealthCheck`

Obbligatorio: no

onUpdateGameSession

Nome della funzione di callback richiamata dal Amazon GameLift Servers servizio per passare un oggetto di sessione di gioco aggiornato al processo del server. Amazon GameLift Servers chiama questa funzione quando è stata elaborata una richiesta di [match backfill](#) per fornire dati aggiornati sul matchmaker. Passa un [GameSession](#) oggetto, uno status update (updateReason) e l'ID del ticket match backfill.

Tipo: `std::function<void(Aws::GameLift::Server::Model::UpdateGameSession)> onUpdateGameSession`

Obbligatorio: no

StartMatchBackfillRequest

Questo tipo di dati viene utilizzato per inviare una richiesta di backfill di abbinamento. Le informazioni vengono comunicate al servizio Amazon GameLift Servers in una chiamata [StartMatchBackfill\(\)](#).

Indice

GameSessionArn

Identificatore univoco della sessione di gioco. L'operazione API [GetGameSessionId\(\)](#) restituisce l'identificatore nel formato ARN.

Tipo: stringa

Obbligatorio: sì

MatchmakingConfigurationArn

Identificatore univoco, sotto forma di un ARN, che il matchmaker utilizza per questa richiesta. Per trovare il matchmaker utilizzato per creare la sessione di gioco originale, esaminare l'oggetto sessione di gioco nella proprietà dei dati del matchmaker. Scopri di più sui dati dei matchmaker in [Word con i dati dei matchmaker](#).

Tipo: stringa

Obbligatorio: sì

Players

Un set di dati che rappresenta tutti i giocatori che sono attualmente impegnati nella sessione di gioco. Il matchmaker utilizza queste informazioni per cercare nuovi giocatori che rappresentano un buon abbinamento per i giocatori attuali. Consulta la Guida di riferimento Amazon GameLift Servers API per una descrizione del formato degli oggetti Player. Per trovare gli attributi dei giocatori e gli incarichi della squadra, cerca nell'oggetto della sessione di gioco, nella proprietà dei dati del matchmaker. IDs Se la latenza viene utilizzata dal matchmaker, raccogliere la latenza aggiornata per la regione attuale e includerla nei dati di ciascun giocatore.

Tipo: std::vectorhttps://docs.aws.amazon.com/gamelift/latest/apireference/API_Player.html
<player>

Obbligatorio: sì

TicketId

Identificatore univoco per un abbinamento o un ticket di richiesta di backfill degli abbinamenti. Se non viene fornito alcun valore qui, ne Amazon GameLift Servers genererà uno sotto forma di UUID. Utilizzare questo identificatore per monitorare lo stato del ticket di backfill degli abbinamenti o annullare la richiesta, se necessario.

▪Tipo: stringa

Obbligatorio: no

StopMatchBackfillRequest

Questo tipo di dati viene utilizzato per annullare una richiesta di backfill di abbinamento. Le informazioni vengono comunicate al servizio Amazon GameLift Servers in una chiamata [StopMatchBackfill\(\)](#).

Indice

GameSessionArn

Identificatore univoco della sessione di gioco associato alla richiesta in fase di annullamento.

Tipo: stringa

Obbligatorio: sì

MatchmakingConfigurationArn

Identificatore univoco del matchmaker a cui è stata inviata questa richiesta.

Tipo: stringa

Obbligatorio: sì

TicketId

Identificatore univoco del ticket di richiesta di backfill da annullare.

Tipo: stringa

Obbligatorio: sì

[SDK per server C++ per Amazon GameLift Servers 4.x -- Tipi di dati](#)

Argomenti

- [AcceptPlayerSession\(\)](#)
- [ActivateGameSession\(\)](#)
- [DescribePlayerSessions\(\)](#)

- [GetGameSessionId\(\)](#)
- [GetInstanceCertificate\(\)](#)
- [GetSdkVersion\(\)](#)
- [GetTerminationTime\(\)](#)
- [InitSDK\(\)](#)
- [ProcessEnding\(\)](#)
- [ProcessReady\(\)](#)
- [ProcessReadyAsync\(\)](#)
- [RemovePlayerSession\(\)](#)
- [StartMatchBackfill\(\)](#)
- [StopMatchBackfill\(\)](#)
- [TerminateGameSession\(\)](#)
- [UpdatePlayerSessionCreationPolicy\(\)](#)
- [Distruggi \(\)](#)

AcceptPlayerSession()

Notifica al Amazon GameLift Servers servizio che un giocatore con l'ID di sessione del giocatore specificato si è connesso al processo del server e deve essere convalidato. Amazon GameLift Servers verifica che l'ID della sessione del giocatore sia valido, ovvero che l'ID giocatore abbia riservato uno slot nella sessione di gioco. Dopo la convalida, Amazon GameLift Servers modifica lo stato dello slot giocatore da RESERVED (RISERVATO) ad ACTIVE (ATTIVO).

Sintassi

```
GenericOutcome AcceptPlayerSession(const std::string& playerSessionId);
```

Parameters

playerSessionId

ID univoco rilasciato dal Amazon GameLift Servers servizio in risposta a una chiamata all'azione dell'API AWS SDK Amazon GameLift Servers. [CreatePlayerSession](#) Il client di gioco fa riferimento a questo ID durante la connessione al processo del server.

Tipo: std::string

Obbligatorio: sì

Valore restituito

Restituisce un risultato generico costituito da successo o fallimento con un messaggio di errore.

Esempio

Questo esempio illustra una funzione per la gestione di una richiesta di connessione, inclusa la convalida e il rifiuto di una sessione giocatore non valida. IDs

```
void ReceiveConnectingPlayerSessionID (Connection& connection, const std::string&
playerSessionId){
    Aws::GameLift::GenericOutcome connectOutcome =
        Aws::GameLift::Server::AcceptPlayerSession(playerSessionId);
    if(connectOutcome.IsSuccess())
    {
        connectionToSessionMap.emplace(connection, playerSessionId);
        connection.Accept();
    }
    else
    {
        connection.Reject(connectOutcome.GetError().GetMessage());
    }
}
```

ActivateGameSession()

Notifica al servizio Amazon GameLift Servers che il processo del server ha avviato una sessione di gioco ed è pronto per ricevere le connessioni dei giocatori. Questa operazione deve essere chiamata come parte della funzione di callback `onStartGameSession()`, dopo il completamento dell'inizializzazione di tutte le sessioni di gioco.

Sintassi

```
GenericOutcome ActivateGameSession();
```

Parameters

Questa operazione non prevede parametri.

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

Questo esempio illustra la chiamata a `ActivateGameSession()` nell'ambito della funzione di callback `onStartGameSession()`.

```
void onStartGameSession(Aws::GameLift::Model::GameSession myGameSession)
{
    // game-specific tasks when starting a new game session, such as loading map
    GenericOutcome outcome = Aws::GameLift::Server::ActivateGameSession();
}
```

DescribePlayerSessions()

Recupera i dati della sessione giocatore, tra cui le impostazioni, i metadati della sessione e i dati dei giocatori. Utilizza questa operazione per ottenere le informazioni per una singola sessione giocatore, per tutte le sessioni giocatore in una sessione di gioco o per tutte le sessioni giocatore associate a un singolo ID giocatore.

Sintassi

```
DescribePlayerSessionsOutcome DescribePlayerSessions (
    const Aws::GameLift::Server::Model::DescribePlayerSessionsRequest
    &describePlayerSessionsRequest);
```

Parameters

describePlayerSessionsRichiesta

Un oggetto [DescribePlayerSessionsRequest](#) che descrive le sessioni giocatore da recuperare.

Obbligatorio: sì

Valore restituito

Se l'esito è positivo, restituisce un oggetto `DescribePlayerSessionsOutcome` contenente un set di oggetti di sessione giocatore corrispondente ai parametri della richiesta. Gli oggetti della sessione Player hanno una struttura identica al tipo di [PlayerSession](#) dati dell'Amazon GameLift ServersAPI AWS SDK.

Esempio

Questo esempio illustra una richiesta per tutte le sessioni giocatore attivamente connesse a una sessione di gioco specificata. Se ometti `NextToken` e imposti il valore `Limit` su 10, Amazon GameLift Servers restituisce i primi 10 record di sessioni giocatore che corrispondono alla richiesta.

```
// Set request parameters
Aws::GameLift::Server::Model::DescribePlayerSessionsRequest request;
request.SetPlayerSessionStatusFilter(Aws::GameLift::Server::Model::PlayerSessionStatusMapper::G
request.SetLimit(10);
request.SetGameSessionId("the game session ID");    // can use GetGameSessionId()

// Call DescribePlayerSessions
Aws::GameLift::DescribePlayerSessionsOutcome playerSessionsOutcome =
    Aws::GameLift::Server::DescribePlayerSessions(request);
```

GetGameSessionId()

Recupera un identificatore univoco della sessione di gioco attualmente ospitata dal processo del server, se il processo del server è attivo. L'identificatore viene restituito nel formato ARN: `arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string>`.

Per i processi inattivi che non sono ancora stati attivati con una sessione di gioco, la chiamata restituisce `Success = True` e `GameSessionId = ""` (una stringa vuota).

Sintassi

```
AwsStringOutcome GetGameSessionId();
```

Parameters

Questa operazione non prevede parametri.

Valore restituito

Se l'esito è positivo, l'ID della sessione di gioco verrà restituito come oggetto `AwsStringOutcome`. Se l'esito è negativo, verrà restituito un messaggio di errore.

Esempio

```
Aws::GameLift::AwsStringOutcome sessionIdOutcome =
    Aws::GameLift::Server::GetGameSessionId();
```

GetInstanceCertificate()

Recupera la posizione del file di un certificato TLS con codifica pem associato alla flotta e alle relative istanze. AWS Certificate Manager genera questo certificato quando crei un nuovo parco veicoli con la configurazione del certificato impostata su GENERATED. Utilizza questo certificato per stabilire una connessione sicura con un client di gioco e per crittografare la comunicazione client/server.

Sintassi

```
GetInstanceCertificateOutcome GetInstanceCertificate();
```

Parameters

Questa operazione non prevede parametri.

Valore restituito

In caso di successo, restituisce un `GetInstanceCertificateOutcome` oggetto contenente la posizione del file di certificato TLS e della catena di certificati della flotta, che sono archiviati nell'istanza. Nell'istanza viene inoltre archiviato un file di certificato radice, estratto dalla catena di certificati. Se l'esito è negativo, verrà restituito un messaggio di errore.

Per ulteriori informazioni sul certificato e sui dati della catena di certificati, consulta [GetCertificate Response Elements](#) nell' AWS Certificate Manager API Reference.

Esempio

```
Aws::GameLift::GetInstanceCertificateOutcome certificateOutcome =  
    Aws::GameLift::Server::GetInstanceCertificate();
```

GetSdkVersion()

Restituisce il numero di versione corrente dell'SDK in uso.

Sintassi

```
AwsStringOutcome GetSdkVersion();
```

Parameters

Questa operazione non prevede parametri.

Valore restituito

Se l'esito è positivo, restituisce la versione corrente dell'SDK come oggetto `AwsStringOutcome`. La stringa restituita include solo il numero di versione (ad esempio «3.1.5»). Se l'esito è negativo, verrà restituito un messaggio di errore.

Esempio

```
Aws::GameLift::AwsStringOutcome SdkVersionOutcome =  
    Aws::GameLift::Server::GetSdkVersion();
```

GetTerminationTime()

Restituisce il tempo di arresto pianificato di un processo del server, se è disponibile un tempo di chiusura. Un processo server esegue questa azione dopo aver ricevuto una `onProcessTerminate()` richiamata dal servizio. Amazon GameLift Servers Amazon GameLift Servers [può effettuare una chiamata `onProcessTerminate\(\)` per i seguenti motivi: \(1\) quando il processo server ha segnalato problemi di salute o non ha risposto Amazon GameLift Servers, \(2\) quando chiude l'istanza durante un evento di scale-down o \(3\) quando un'istanza viene interrotta a causa di un'interruzione Spot.](#)

Se il processo ha ricevuto un `onProcessTerminate()` callback, il valore restituito è il tempo di terminazione stimato. Se il processo non ha ricevuto una `onProcessTerminate()` richiamata, viene restituito un messaggio di errore. Ulteriori informazioni sull'[arresto di un processo del server](#).

Sintassi

```
AwsLongOutcome GetTerminationTime();
```

Parameters

Questa operazione non prevede parametri.

Valore restituito

In caso di successo, restituisce l'ora di terminazione come oggetto `AwsLongOutcome`. Il valore è l'ora di terminazione, espressa in segni di spunta trascorsi dalle 00:01 00:00:00. Ad esempio, il valore data/ora 2020-09-13 12:26:40 -000Z è uguale a 637355968000000000 ticks. Se non è disponibile alcun orario di terminazione, restituisce un messaggio di errore.

Esempio

```
Aws::GameLift::AwsLongOutcome TermTimeOutcome =  
    Aws::GameLift::Server::GetTerminationTime();
```

InitSDK()

Inizializza l'SDK Amazon GameLift Servers. Questo metodo deve essere chiamato all'avvio, prima che avvenga qualsiasi altra inizializzazione correlata a Amazon GameLift Servers.

Sintassi

```
InitSDKOutcome InitSDK();
```

Parameters

Questa operazione non prevede parametri.

Valore restituito

In caso di successo, restituisce un `InitSdkOutcome` oggetto che indica che il processo del server è pronto per la chiamata [ProcessReady\(\)](#).

Esempio

```
Aws::GameLift::Server::InitSDKOutcome initOutcome =  
    Aws::GameLift::Server::InitSDK();
```

ProcessEnding()

Avvisa il servizio Amazon GameLift Servers che il processo del server è in fase di arresto. Questo metodo deve essere richiamato dopo tutte le altre attività di pulizia, tra cui l'arresto di tutte le sessioni di gioco attive. Questo metodo deve uscire con il codice 0; un codice di uscita diverso da zero genera un messaggio di evento che indica che il processo non è terminato correttamente.

Una volta terminato il metodo con un codice 0, è possibile terminare il processo con un codice di uscita corretto. È inoltre possibile uscire dal processo con un codice di errore. Se si esce con un codice di errore, l'evento fleet indicherà che il processo è terminato in modo anomalo (`SERVER_PROCESS_TERMINATED_UNHEALTHY`).

Sintassi

```
GenericOutcome ProcessEnding();
```

Parameters

Questa operazione non prevede parametri.

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

```
Aws::GameLift::GenericOutcome outcome = Aws::GameLift::Server::ProcessEnding();
if (outcome.Success)
    exit(0); // exit with success
// otherwise, exit with error code
exit(errorCode);
```

ProcessReady()

Notifica al servizio Amazon GameLift Servers che il processo del server è pronto per l'hosting delle sessioni di gioco. Chiama questo metodo dopo aver richiamato [InitSDK\(\)](#) e completato con successo le attività di configurazione necessarie prima che il processo del server possa ospitare una sessione di gioco. Questo metodo deve essere chiamato solo una volta per processo.

Questa chiamata è sincrona. Per effettuare una chiamata asincrona, utilizza [ProcessReadyAsync\(\)](#). Per ulteriori dettagli, consulta [Inizializza il processo del server](#).

Sintassi

```
GenericOutcome ProcessReady(
    const Aws::GameLift::Server::ProcessParameters &processParameters);
```

Parameters

processParameters

Un oggetto [ProcessParameters](#) che comunica le informazioni seguenti sul processo del server:

- Nomi di metodi di callback, implementati nel codice del server di gioco, che il servizio Amazon GameLift Servers richiama per comunicare con il processo del server.
- Numero di porta sulla quale è in ascolto il processo del server.
- Percorso verso qualsiasi file specifico di una sessione di gioco che Amazon GameLift Servers deve acquisire e archiviare.

Obbligatorio: sì

Valore restituito

Restituisce un risultato generico costituito da successo o fallimento con un messaggio di errore.

Esempio

Questo esempio illustra le implementazioni della funzione di chiamata e callback [ProcessReady\(\)](#).

```
// Set parameters and call ProcessReady
std::string serverLog("serverOut.log");           // Example of a log file written by the
game server
std::vector<std::string> logPaths;
logPaths.push_back(serverLog);

int listenPort = 9339;

Aws::GameLift::Server::ProcessParameters processReadyParameter =
    Aws::GameLift::Server::ProcessParameters(
        std::bind(&Server::onStartGameSession, this, std::placeholders::_1),
        std::bind(&Server::onProcessTerminate, this),
        std::bind(&Server::OnHealthCheck, this),
        std::bind(&Server::OnUpdateGameSession, this),
        listenPort,
        Aws::GameLift::Server::LogParameters(logPaths));

Aws::GameLift::GenericOutcome outcome =
    Aws::GameLift::Server::ProcessReady(processReadyParameter);

// Implement callback functions
void Server::onStartGameSession(Aws::GameLift::Model::GameSession myGameSession)
{
    // game-specific tasks when starting a new game session, such as loading map
    GenericOutcome outcome =
        Aws::GameLift::Server::ActivateGameSession (maxPlayers);
```

```
}

void Server::onProcessTerminate()
{
    // game-specific tasks required to gracefully shut down a game session,
    // such as notifying players, preserving game state data, and other cleanup
    GenericOutcome outcome = Aws::GameLift::Server::ProcessEnding();
}

bool Server::onHealthCheck()
{
    bool health;
    // complete health evaluation within 60 seconds and set health
    return health;
}
```

ProcessReadyAsync()

Notifica al servizio Amazon GameLift Servers che il processo del server è pronto per l'hosting delle sessioni di gioco. Questo metodo deve essere chiamato quando il processo del server è pronto per l'hosting di una sessione di gioco. I parametri specificano i nomi delle funzioni di callback che Amazon GameLift Servers deve chiamare in determinate condizioni. Il codice del server di gioco deve implementare queste funzioni.

Questa chiamata è asincrona. Per effettuare una chiamata sincrona, utilizza [ProcessReady\(\)](#). Per ulteriori dettagli, consulta [Inizializza il processo del server](#).

Sintassi

```
GenericOutcomeCallable ProcessReadyAsync(
    const Aws::GameLift::Server::ProcessParameters &processParameters);
```

Parameters

processParameters

Un oggetto [ProcessParameters](#) che comunica le informazioni seguenti sul processo del server:

- Nomi di metodi di callback, implementati nel codice del server di gioco, che il servizio Amazon GameLift Servers richiama per comunicare con il processo del server.
- Numero di porta sulla quale è in ascolto il processo del server.

- Percorso verso qualsiasi file specifico di una sessione di gioco che Amazon GameLift Servers deve acquisire e archiviare.

Obbligatorio: sì

Valore restituito

Restituisce un risultato generico costituito da esito positivo o negativo con un messaggio di errore.

Esempio

```
// Set parameters and call ProcessReady
std::string serverLog("serverOut.log");           // This is an example of a log file
                                                    written by the game server
std::vector<std::string> logPaths;
logPaths.push_back(serverLog);

int listenPort = 9339;

Aws::GameLift::Server::ProcessParameters processReadyParameter =
    Aws::GameLift::Server::ProcessParameters(
        std::bind(&Server::onStartGameSession, this, std::placeholders::_1),
        std::bind(&Server::onProcessTerminate, this),
        std::bind(&Server::OnHealthCheck, this),
        std::bind(&Server::OnUpdateGameSession, this),
        listenPort,
        Aws::GameLift::Server::LogParameters(logPaths));

Aws::GameLift::GenericOutcomeCallable outcome =
    Aws::GameLift::Server::ProcessReadyAsync(processReadyParameter);

// Implement callback functions
void onStartGameSession(Aws::GameLift::Model::GameSession myGameSession)
{
    // game-specific tasks when starting a new game session, such as loading map
    GenericOutcome outcome = Aws::GameLift::Server::ActivateGameSession (maxPlayers);
}

void onProcessTerminate()
{
    // game-specific tasks required to gracefully shut down a game session,
    // such as notifying players, preserving game state data, and other cleanup
    GenericOutcome outcome = Aws::GameLift::Server::ProcessEnding();
}
```



```
}

bool onHealthCheck()
{
    // perform health evaluation and complete within 60 seconds
    return health;
}
```

RemovePlayerSession()

Notifica al servizio Amazon GameLift Servers che un giocatore con l'ID della sessione giocatore specificato si è disconnesso dal processo del server. In risposta, Amazon GameLift Servers modifica lo slot giocatore su disponibile, consentendone l'assegnazione a un nuovo giocatore.

Sintassi

```
GenericOutcome RemovePlayerSession(
    const std::string& playerId);
```

Parameters

playerSessionId

ID univoco rilasciato dal Amazon GameLift Servers servizio in risposta a una chiamata all'azione [CreatePlayerSession](#) dell'Amazon GameLift Servers API AWS SDK. Il client di gioco fa riferimento a questo ID durante la connessione al processo del server.

Tipo: `std::string`

Obbligatorio: sì

Valore restituito

Restituisce un risultato generico costituito da successo o fallimento con un messaggio di errore.

Esempio

```
Aws::GameLift::GenericOutcome disconnectOutcome =
    Aws::GameLift::Server::RemovePlayerSession(playerSessionId);
```

StartMatchBackfill()

Invia una richiesta per trovare nuovi giocatori per gli slot aperti in una sessione di gioco creata con FlexMatch. Vedi anche l'azione AWS SDK [StartMatchBackfill\(\)](#). Con questa operazione, è possibile avviare le richieste di backfill degli abbinamenti da un processo del server di gioco che ospita la sessione di gioco. Scopri di più sulla funzione di [FlexMatchriempimento](#).

Questa operazione è asincrona. Se i nuovi giocatori vengono abbinati con successo, il Amazon GameLift Servers servizio fornisce dati aggiornati del matchmaker richiamando la funzione di callback. `OnUpdateGameSession()`

Un processo del server può avere un solo backfill degli abbinamenti attivo alla volta. Per inviare una nuova richiesta, chiama prima [StopMatchBackfill\(\)](#) per annullare la richiesta originale.

Sintassi

```
StartMatchBackfillOutcome StartMatchBackfill (
    const Aws::GameLift::Server::Model::StartMatchBackfillRequest
    &startBackfillRequest);
```

Parameters

StartMatchBackfillRequest

Un oggetto [StartMatchBackfillRequest](#) che comunica le informazioni seguenti:

- ID del ticket da assegnare alla richiesta di backfill. Questa informazione è facoltativa; se l'ID non viene fornito, Amazon GameLift Servers ne creerà automaticamente uno.
- Matchmaker a cui inviare la richiesta. L'ARN di configurazione completo è obbligatorio. Questo valore può essere acquisito dai dati del matchmaker della sessione di gioco.
- ID della sessione di gioco che è in fase di backfilling.
- Dati di abbinamento disponibili per i giocatori correnti della sessione di gioco.

Obbligatorio: sì

Valore restituito

Restituisce un `StartMatchBackfillOutcome` oggetto con il match backfill ticket o un errore con un messaggio di errore. [Lo stato del ticket può essere monitorato utilizzando l'azione AWS DescribeMatchmaking SDK \(\)](#).

Esempio

```
// Build a backfill request
std::vector<Player> players;
Aws::GameLift::Server::Model::StartMatchBackfillRequest startBackfillRequest;
startBackfillRequest.SetTicketId("a ticket ID");
    //optional, autogenerated if not provided
startBackfillRequest.SetMatchmakingConfigurationArn("the matchmaker configuration
    ARN"); //from the game session matchmaker data
startBackfillRequest.SetGameSessionArn("the game session ARN");
    // can use GetGameSessionId()
startBackfillRequest.SetPlayers(players);
    //from the game session matchmaker data

// Send backfill request
Aws::GameLift::StartMatchBackfillOutcome backfillOutcome =
    Aws::GameLift::Server::StartMatchBackfill(startBackfillRequest);

// Implement callback function for backfill
void Server::OnUpdateGameSession(Aws::GameLift::Server::Model::GameSession gameSession,
    Aws::GameLift::Server::Model::UpdateReason updateReason, std::string backfillTicketId)
{
    // handle status messages
    // perform game-specific tasks to prep for newly matched players
}
```

StopMatchBackfill()

Annulla una richiesta di backfill degli abbinamenti attiva creata con [StartMatchBackfill\(\)](#). [Vedi anche l'azione AWS StopMatchmaking SDK \(\)](#). Scopri di più sulla funzione di [FlexMatchriempimento](#).

Sintassi

```
GenericOutcome StopMatchBackfill (
    const Aws::GameLift::Server::Model::StopMatchBackfillRequest &stopBackfillRequest);
```

Parameters

StopMatchBackfillRequest

Un oggetto [StopMatchBackfillRequest](#) che identifica il ticket di abbinamento da annullare:

- ID del ticket assegnato alla richiesta di backfill in fase di annullamento

- matchmaker a cui è stata inviata la richiesta di backfill
- sessione di gioco associata alla richiesta di backfill

Obbligatorio: sì

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

```
// Set backfill stop request parameters

Aws::GameLift::Server::Model::StopMatchBackfillRequest stopBackfillRequest;
stopBackfillRequest.SetTicketId("the ticket ID");
stopBackfillRequest.SetGameSessionArn("the game session ARN");
    // can use GetGameSessionId()
stopBackfillRequest.SetMatchmakingConfigurationArn("the matchmaker configuration ARN");
    // from the game session matchmaker data

Aws::GameLift::GenericOutcome stopBackfillOutcome =
    Aws::GameLift::Server::StopMatchBackfillRequest(stopBackfillRequest);
```

TerminateGameSession()

Questo metodo è obsoleto con la versione 4.0.1. Invece, il processo del server dovrebbe richiamare [ProcessEnding\(\)](#) dopo la fine di una sessione di gioco.

Notifica al Amazon GameLift Servers servizio che il processo del server ha terminato la sessione di gioco corrente. Questa azione viene richiamata quando il processo del server rimane attivo e pronto per ospitare una nuova sessione di gioco. Dovrebbe essere chiamata solo dopo il completamento della procedura di interruzione della sessione di gioco, perché segnala Amazon GameLift Servers che il processo del server è immediatamente disponibile per ospitare una nuova sessione di gioco.

Questa azione non viene eseguita se il processo del server verrà chiuso dopo l'interruzione della sessione di gioco. Invece, chiama [ProcessEnding\(\)](#) per segnalare che sia la sessione di gioco che il processo sul server stanno terminando.

Sintassi

```
GenericOutcome TerminateGameSession();
```

Parameters

Questa operazione non prevede parametri.

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

UpdatePlayerSessionCreationPolicy()

Aggiorna la capacità della sessione di gioco corrente di accettare nuove sessioni giocatore. Una sessione di gioco può essere configurata per accettare o rifiutare tutte le nuove sessioni giocatore.

Vedi anche l'azione AWS SDK [UpdateGameSession\(\)](#).

Sintassi

```
GenericOutcome UpdatePlayerSessionCreationPolicy(  
    Aws::GameLift::Model::PlayerSessionCreationPolicy newPlayerSessionPolicy);
```

Parameters

newPlayerSessionPolitica

Valore della stringa che indica se la sessione di gioco accetta nuovi giocatori.

Tipo: `Aws::GameLift::Model::PlayerSessionCreationPolicy` enum. I valori validi includono:

- `ACCEPT_ALL`: accetta tutte le nuove sessioni giocatore.
- `DENY_ALL`: rifiuta tutte le nuove sessioni giocatore.

Obbligatorio: sì

Valore restituito

Restituisce un risultato generico costituito da successo o fallimento con un messaggio di errore.

Esempio

Questo esempio definisce la policy di partecipazione alla sessione di gioco corrente per accettare tutti i giocatori.

```
Aws::GameLift::GenericOutcome outcome =  
    Aws::GameLift::Server::UpdatePlayerSessionCreationPolicy(Aws::GameLift::Model::PlayerSessionCr
```

Distruggi ()

Pulisce la memoria allocata da `initSDK ()` durante l'inizializzazione del server di gioco. Utilizza questo metodo dopo aver terminato un processo del server di gioco per evitare di sprecare memoria del server.

Sintassi

```
GenericOutcome Aws::GameLift::Server::Destroy();
```

Parameters

Non ci sono parametri.

Valore restituito

Restituisce un risultato generico costituito da successo o fallimento con un messaggio di errore.

Esempio

Questo esempio pulisce la memoria allocata da `InitSDK` al termine di un processo sul server di gioco.

```
if (Aws::GameLift::Server::ProcessEnding().IsSuccess()) {  
    Aws::GameLift::Server::Destroy();  
    exit(0);  
}
```

SDK del server C# per Amazon GameLift Servers 4.x -- Azioni

Usa il riferimento all'SDK del server per integrare il tuo gioco multiplayer con cui ospitarlo. Amazon GameLift Servers Per indicazioni sul processo di integrazione, consulta [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#).

Note

Questo riferimento si riferisce a una versione precedente del server SDK for Amazon GameLift Servers. Per la versione più recente, consulta [Server C# SDK 5.x per -- Azioni Amazon GameLift Servers](#).

SDK del server C# per Amazon GameLift Servers 4.x -- Tipi di dati

Usa il riferimento all'SDK del server per integrare il tuo gioco multiplayer con cui ospitarlo. Amazon GameLift Servers Per indicazioni sul processo di integrazione, consulta [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#).

Note

Questo riferimento si riferisce a una versione precedente del server SDK for Amazon GameLift Servers. Per la versione più recente, consulta [Server C# SDK 5.x per Amazon GameLift Servers -- Tipi di dati](#).

[SDK del server C# per Amazon GameLift Servers 4.x -- Azioni](#)

Argomenti

- [LogParameters](#)
- [DescribePlayerSessionsRequest](#)
- [ProcessParameters](#)
- [StopMatchBackfillRequest](#)

LogParameters

Questo tipo di dati viene utilizzato per identificare i file generati durante una sessione di gioco che Amazon GameLift Servers caricherà e memorizzerà al termine di questa fase. Queste informazioni vengono comunicate al servizio Amazon GameLift Servers in una chiamata [ProcessReady\(\)](#).

Indice

logPaths

Elenco dei percorsi di directory ai file di log del server di gioco che Amazon GameLift Servers archivia per l'accesso futuro. Questi file vengono generati da un processo server durante ogni sessione di gioco; i percorsi dei file e i nomi sono definiti nel server di gioco e memorizzati nella directory root build di gioco. I percorsi di registro devono essere assoluti. Ad esempio, se la build di gioco archivia i log della sessione di gioco in un percorso come MyGame\sessionlogs\, il percorso di log è c:\game\MyGame\sessionLogs (su un'istanza Windows) o /local/game/MyGame/sessionLogs (su un'istanza Linux).

Tipo: List<String>

Obbligatorio: no

DescribePlayerSessionsRequest

Questo tipo di dati viene utilizzato per specificare quale sessione del giocatore recuperare. Può essere utilizzato in diversi modi: (1) fornire `PlayerSessionId` a per richiedere una sessione di gioco specifica; (2) fornire `GameSessionId` a per richiedere tutte le sessioni dei giocatori nella sessione di gioco specificata; o (3) fornire `PlayerId` a per richiedere tutte le sessioni di gioco per il giocatore specificato. Per le raccolte di grandi dimensioni delle sessioni giocatore, utilizzare i parametri di paginazione per recuperare i risultati in pagine sequenziali.

Indice

GameSessionId

Identificatore univoco della sessione di gioco. Utilizzare questo parametro per richiedere tutte le sessioni giocatore per la sessione di gioco specificata. Il formato dell'ID della sessione di gioco è il seguente: `arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string>`. Il valore di `<ID string>` corrisponde a una stringa ID personalizzata o (se ne è stata specificata una al momento della creazione della sessione di gioco) a una stringa generata.

─Tipo: stringa

Obbligatorio: no

Limite

Numero massimo di risultati da restituire. Utilizzate questo parametro con `NextToken` per ottenere risultati sotto forma di set di pagine sequenziali. Se viene specificato un ID sessione giocatore, questo parametro verrà ignorato.

Tipo: Integer

Obbligatorio: no

NextToken

Token che indica l'inizio della pagina sequenziale successiva relativa ai risultati. Utilizzare il token restituito con una chiamata precedente a questa operazione. Per specificare l'inizio del set di risultati, non specificare un valore. Se viene specificato un ID sessione giocatore, questo parametro verrà ignorato.

■Tipo: stringa

Obbligatorio: no

PlayerId

Identificatore univoco per un giocatore. IDs I giocatori sono definiti dallo sviluppatore. Per informazioni, consulta [Genera giocatore IDs](#).

■Tipo: stringa

Obbligatorio: no

PlayerSessionId

Identificatore univoco della sessione giocatore.

■Tipo: stringa

Obbligatorio: no

PlayerSessionStatusFilter

Stato sessione giocatore su cui filtrare i risultati. Tra gli stati sessione giocatore possibili sono inclusi i seguenti:

- RISERVATO — La richiesta di sessione del giocatore è stata ricevuta, ma il giocatore non si è ancora connesso al server. Il processo and/or è stato convalidato.
- ACTIVE (ATTIVO) - Il giocatore è stato convalidato dal processo del server ed è attualmente collegato.
- COMPLETED (COMPLETATO) - La connessione del giocatore è stata interrotta.
- TIMEDOUT — È stata ricevuta una richiesta di sessione di gioco, ma il giocatore che non si and/or è connesso non è stata convalidata entro il limite di timeout (60 secondi).

■Tipo: stringa

Obbligatorio: no

ProcessParameters

Questo tipo di dati contiene il set di parametri inviato al servizio Amazon GameLift Servers in una chiamata [ProcessReady\(\)](#).

Indice

port

Numero di porta a cui il processo del server rimarrà in attesa di nuove connessioni dei giocatori. Il valore deve rientrare nella gamma di porte configurate per qualsiasi parco istanze che distribuisce questa build del server di gioco. Questo numero di porta è incluso nella sessione di gioco e negli oggetti della sessione del giocatore utilizzati dalle sessioni di gioco per la connessione a un processo server.

Tipo: numero intero

Obbligatorio: sì

logParameters

Un oggetto con un elenco di percorsi delle directory per i file log delle sessioni di gioco.

Tipo: `Aws::GameLift::Server::LogParameters`

Obbligatorio: sì

onStartGameSessione

Nome della funzione di callback richiamata dal Amazon GameLift Servers servizio per attivare una nuova sessione di gioco. Amazon GameLift Servers chiama questa funzione in risposta alla richiesta del client. [CreateGameSession](#) La funzione di callback accetta un [GameSession](#) oggetto (definito nel Amazon GameLift ServersService API Reference).

Tipo: `void OnStartGameSessionDelegate(GameSession gameSession)`

Obbligatorio: sì

onProcessTerminate

Nome della funzione di callback richiamata dal Amazon GameLift Servers servizio per forzare la chiusura del processo del server. Dopo aver richiamato questa funzione, Amazon GameLift Servers attende per cinque minuti l'arresto del processo del server e risponde con una chiamata [ProcessEnding\(\)](#) prima di arrestare il processo del server.

Tipo: `void OnProcessTerminateDelegate()`

Obbligatorio: sì

onHealthCheck

Nome della funzione di callback richiamata dal Amazon GameLift Servers servizio per richiedere un rapporto sullo stato di salute del processo del server. Amazon GameLift Servers chiama questa funzione ogni 60 secondi. Dopo aver richiamato questa funzione, Amazon GameLift Servers attende per 60 secondi l'arrivo di una risposta e, se non ne riceve alcuna, registra il processo del server come non integro.

Tipo: `bool OnHealthCheckDelegate()`

Obbligatorio: sì

onUpdateGameSessione

Nome della funzione di callback richiamata dal Amazon GameLift Servers servizio per passare un oggetto di sessione di gioco aggiornato al processo del server. Amazon GameLift Servers chiama questa funzione quando è stata elaborata una richiesta di [match backfill](#) per fornire dati aggiornati sul matchmaker. Passa un [GameSession](#) oggetto, uno status update (`updateReason`) e l'ID del ticket match backfill.

Tipo: `void OnUpdateGameSessionDelegate (UpdateGameSession
updateGameSession)`

Obbligatorio: no

StartMatchBackfillRequest

Questo tipo di dati viene utilizzato per inviare una richiesta di backfill di abbinamento. Le informazioni vengono comunicate al servizio Amazon GameLift Servers in una chiamata [StartMatchBackfill\(\)](#).

Indice

GameSessionArn

Identificatore univoco della sessione di gioco. Il metodo SDK [GetGameSessionId\(\)](#) restituisce l'identificatore in formato ARN.

Tipo: stringa

Obbligatorio: sì

MatchmakingConfigurationArn

Identificatore univoco, sotto forma di un ARN, che il matchmaker utilizza per questa richiesta. Per trovare il matchmaker utilizzato per creare la sessione di gioco originale, esaminare l'oggetto sessione di gioco nella proprietà dei dati del matchmaker. Scopri di più sui dati dei matchmaker in [Work with matchmaker data](#).

Tipo: stringa

Obbligatorio: sì

Players

Un set di dati che rappresenta tutti i giocatori che sono attualmente impegnati nella sessione di gioco. Il matchmaker utilizza queste informazioni per cercare nuovi giocatori che rappresentano un buon abbinamento per i giocatori attuali. Consulta la Guida di riferimento Amazon GameLift Servers API per una descrizione del formato degli oggetti Player. Per trovare gli attributi dei giocatori e gli incarichi della squadra, cerca nell'oggetto della sessione di gioco, nella proprietà dei dati del matchmaker. IDs Se la latenza viene utilizzata dal matchmaker, raccogliere la latenza aggiornata per la regione attuale e includerla nei dati di ciascun giocatore.

Tipo: [Giocatore](#)[]

Obbligatorio: sì

TicketId

Identificatore univoco per un abbinamento o un ticket di richiesta di backfill degli abbinamenti. Se non viene fornito alcun valore qui, ne Amazon GameLift Servers genererà uno sotto forma di UUID. Utilizzare questo identificatore per monitorare lo stato del ticket di backfill degli abbinamenti o annullare la richiesta, se necessario.

─Tipo: stringa

Obbligatorio: no

StopMatchBackfillRequest

Questo tipo di dati viene utilizzato per annullare una richiesta di backfill di abbinamento. Le informazioni vengono comunicate al servizio Amazon GameLift Servers in una chiamata [StopMatchBackfill\(\)](#).

Indice

GameSessionArn

Identificatore univoco della sessione di gioco associato alla richiesta in fase di annullamento.

Tipo: stringa

Obbligatorio: sì

MatchmakingConfigurationArn

Identificatore univoco del matchmaker a cui è stata inviata questa richiesta.

Tipo: stringa

Obbligatorio: sì

TicketId

Identificatore univoco del ticket di richiesta di backfill da annullare.

Tipo: stringa

Obbligatorio: sì

[SDK del server C# per Amazon GameLift Servers 4.x -- Tipi di dati](#)

Argomenti

- [AcceptPlayerSession\(\)](#)
- [ActivateGameSession\(\)](#)
- [DescribePlayerSessions\(\)](#)
- [GetGameSessionId\(\)](#)
- [GetInstanceCertificate\(\)](#)
- [GetSdkVersion\(\)](#)
- [GetTerminationTime\(\)](#)
- [InitSDK\(\)](#)
- [ProcessEnding\(\)](#)

- [ProcessReady\(\)](#)
- [RemovePlayerSession\(\)](#)
- [StartMatchBackfill\(\)](#)
- [StopMatchBackfill\(\)](#)
- [TerminateGameSession\(\)](#)
- [UpdatePlayerSessionCreationPolicy\(\)](#)

AcceptPlayerSession()

Notifica al Amazon GameLift Servers servizio che un giocatore con l'ID di sessione del giocatore specificato si è connesso al processo del server e deve essere convalidato. Amazon GameLift Servers verifica che l'ID della sessione del giocatore sia valido, ovvero che l'ID giocatore abbia riservato uno slot nella sessione di gioco. Dopo la convalida, Amazon GameLift Servers modifica lo stato dello slot giocatore da RESERVED (RISERVATO) ad ACTIVE (ATTIVO).

Sintassi

```
GenericOutcome AcceptPlayerSession(String playerId)
```

Parameters

playerSessionId

ID univoco rilasciato Amazon GameLift Servers quando viene creata una nuova sessione giocatore. Un ID di sessione del giocatore viene specificato in un `PlayerSession` oggetto, che viene restituito in risposta a una chiamata del client alle azioni GameLift API [StartGameSessionPlacement](#), [CreateGameSession](#), [DescribeGameSessionPlacement](#), o [DescribePlayerSessions](#).

Tipo: stringa

Obbligatorio: sì

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

Questo esempio illustra una funzione per la gestione di una richiesta di connessione, inclusa la convalida e il rifiuto di una sessione giocatore non valida. IDs

```
void ReceiveConnectingPlayerSessionID (Connection connection, String playerId){
    var acceptPlayerSessionOutcome =
    GameLiftServerAPI.AcceptPlayerSession(playerSessionId);
    if(acceptPlayerSessionOutcome.Success)
    {
        connectionToSessionMap.emplace(connection, playerId);
        connection.Accept();
    }
    else
    {
        connection.Reject(acceptPlayerSessionOutcome.Error.ErrorMessage);    }
}
```

ActivateGameSession()

Notifica al servizio Amazon GameLift Servers che il processo del server ha attivato una sessione di gioco ed è pronto per ricevere le connessioni dei giocatori. Questa operazione deve essere chiamata come parte della funzione di callback `onStartGameSession()`, dopo il completamento dell'inizializzazione di tutte le sessioni di gioco.

Sintassi

```
GenericOutcome ActivateGameSession()
```

Parameters

Questa operazione non prevede parametri.

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

Questo esempio illustra la chiamata a `ActivateGameSession()` nell'ambito della funzione delegata `onStartGameSession()`.

```
void OnStartGameSession(GameSession gameSession)
{
    // game-specific tasks when starting a new game session, such as loading map

    // When ready to receive players
    var activateGameSessionOutcome = GameLiftServerAPI.ActivateGameSession();
}
```

DescribePlayerSessions()

Recupera i dati della sessione giocatore, tra cui le impostazioni, i metadati della sessione e i dati dei giocatori. Utilizza questa operazione per ottenere le informazioni per una singola sessione giocatore, per tutte le sessioni giocatore in una sessione di gioco o per tutte le sessioni giocatore associate a un singolo ID giocatore.

Sintassi

```
DescribePlayerSessionsOutcome DescribePlayerSessions(DescribePlayerSessionsRequest
describePlayerSessionsRequest)
```

Parameters

describePlayerSessionsRichiesta

Un oggetto [DescribePlayerSessionsRequest](#) che descrive le sessioni giocatore da recuperare.

Obbligatorio: sì

Valore restituito

Se l'esito è positivo, restituisce un oggetto `DescribePlayerSessionsOutcome` contenente un set di oggetti di sessione giocatore corrispondente ai parametri della richiesta. Gli oggetti della sessione Player hanno una struttura identica al tipo di [PlayerSession](#) dati dell'Amazon GameLift ServersAPI AWS SDK.

Esempio

Questo esempio illustra una richiesta per tutte le sessioni giocatore attivamente connesse a una sessione di gioco specificata. Omettendo `NextToken` impostando il valore `Limite` su 10, Amazon

GameLift Servers verranno restituiti i record delle prime 10 sessioni di gioco corrispondenti alla richiesta.

```
// Set request parameters
var describePlayerSessionsRequest = new
    Aws.GameLift.Server.Model.DescribePlayerSessionsRequest()
{
    GameSessionId = GameLiftServerAPI.GetGameSessionId().Result,    //gets the ID for
    the current game session
    Limit = 10,
    PlayerSessionStatusFilter =
    PlayerSessionStatusMapper.GetNameForPlayerSessionStatus(PlayerSessionStatus.ACTIVE)
};
// Call DescribePlayerSessions
Aws::GameLift::DescribePlayerSessionsOutcome playerSessionsOutcome =
    Aws::GameLift::Server::Model::DescribePlayerSessions(describePlayerSessionRequest);
```

GetGameSessionId()

Recupera l'ID della sessione di gioco attualmente ospitata dal processo del server, se il processo del server è attivo.

Per i processi inattivi che non sono ancora stati attivati con una sessione di gioco, la chiamata restituisce `Success = True` e `GameSessionId = ""` (una stringa vuota).

Sintassi

```
AwsStringOutcome GetGameSessionId()
```

Parameters

Questa operazione non prevede parametri.

Valore restituito

Se l'esito è positivo, l'ID della sessione di gioco verrà restituito come oggetto `AwsStringOutcome`. Se l'esito è negativo, verrà restituito un messaggio di errore.

Esempio

```
var getGameSessionIdOutcome = GameLiftServerAPI.GetGameSessionId();
```

GetInstanceCertificate()

Recupera la posizione del file di un certificato TLS con codifica pem associato alla flotta e alle relative istanze. AWS Certificate Manager genera questo certificato quando crei un nuovo parco veicoli con la configurazione del certificato impostata su GENERATED. Utilizza questo certificato per stabilire una connessione sicura con un client di gioco e per crittografare la comunicazione client/server.

Sintassi

```
GetInstanceCertificateOutcome GetInstanceCertificate();
```

Parameters

Questa operazione non prevede parametri.

Valore restituito

In caso di successo, restituisce un `GetInstanceCertificateOutcome` oggetto contenente la posizione del file di certificato TLS e della catena di certificati della flotta, che sono archiviati nell'istanza. Nell'istanza viene inoltre archiviato un file di certificato radice, estratto dalla catena di certificati. Se l'esito è negativo, verrà restituito un messaggio di errore.

Per ulteriori informazioni sul certificato e sui dati della catena di certificati, consulta [GetCertificate Response Elements](#) nell' AWS Certificate Manager API Reference.

Esempio

```
var getInstanceCertificateOutcome = GameLiftServerAPI.GetInstanceCertificate();
```

GetSdkVersion()

Restituisce il numero di versione corrente dell'SDK integrato nel processo del server.

Sintassi

```
AwsStringOutcome GetSdkVersion()
```

Parameters

Questa operazione non prevede parametri.

Valore restituito

Se l'esito è positivo, restituisce la versione corrente dell'SDK come oggetto `AwsStringOutcome`. La stringa restituita include solo il numero di versione (ad esempio «3.1.5»). Se l'esito è negativo, verrà restituito un messaggio di errore.

Esempio

```
var getSdkVersionOutcome = GameLiftServerAPI.GetSdkVersion();
```

GetTerminationTime()

Restituisce il tempo di arresto pianificato di un processo del server, se è disponibile un tempo di chiusura. Un processo server esegue questa azione dopo aver ricevuto una `onProcessTerminate()` richiamata dal servizio. Amazon GameLift Servers [Amazon GameLift Servers può effettuare una chiamata `onProcessTerminate\(\)` per i seguenti motivi: \(1\) per problemi di salute \(il processo del server ha segnalato lo stato della porta o non ha risposto Amazon GameLift Servers, \(2\) quando termina l'istanza durante un evento di scale-down o \(3\) quando un'istanza viene terminata a causa di un'interruzione dell'istanza spot.](#)

Se il processo ha ricevuto una `onProcessTerminate()` richiamata, il valore restituito è il tempo di terminazione stimato. Se il processo non ha ricevuto una `onProcessTerminate()` richiamata, viene restituito un messaggio di errore. Ulteriori informazioni sull'[arresto di un processo del server](#).

Sintassi

```
AwsDateTimeOutcome GetTerminationTime()
```

Parameters

Questa operazione non prevede parametri.

Valore restituito

In caso di successo, restituisce l'ora di terminazione come oggetto `AwsDateTimeOutcome`. Il valore è l'ora di terminazione, espressa in segni di spunta trascorsi dalle 00:01 00:00:00. Ad esempio, il valore data/ora 2020-09-13 12:26:40 -000Z è uguale a 637355968000000000 ticks. Se non è disponibile alcun orario di terminazione, restituisce un messaggio di errore.

Esempio

```
var getTerminationTimeOutcome = GameLiftServerAPI.GetTerminationTime();
```

InitSDK()

Inizializza l'SDK Amazon GameLift Servers. Questo metodo deve essere chiamato all'avvio, prima che avvenga qualsiasi altra inizializzazione correlata a Amazon GameLift Servers.

Sintassi

```
InitSDKOutcome InitSDK()
```

Parameters

Questa operazione non prevede parametri.

Valore restituito

In caso di successo, restituisce un `InitSdkOutcome` oggetto che indica che il processo del server è pronto per la chiamata [ProcessReady\(\)](#).

Esempio

```
var initSDKOutcome = GameLiftServerAPI.InitSDK();
```

ProcessEnding()

Avvisa il servizio Amazon GameLift Servers che il processo del server è in fase di arresto. Questo metodo deve essere richiamato dopo tutte le altre attività di pulizia, tra cui l'arresto di tutte le sessioni di gioco attive. Questo metodo deve uscire con il codice 0; un codice di uscita diverso da zero genera un messaggio di evento che indica che il processo non è terminato correttamente.

Una volta terminato il metodo con un codice 0, è possibile terminare il processo con un codice di uscita corretto. È inoltre possibile uscire dal processo con un codice di errore. Se si esce con un codice di errore, l'evento fleet indicherà che il processo è terminato in modo anomalo (`SERVER_PROCESS_TERMINATED_UNHEALTHY`).

Sintassi

```
GenericOutcome ProcessEnding()
```

Parameters

Questa operazione non prevede parametri.

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

```
var processEndingOutcome = GameLiftServerAPI.ProcessEnding();
if (processReadyOutcome.Success)
    Environment.Exit(0);
// otherwise, exit with error code
Environment.Exit(errorCode);
```

ProcessReady()

Notifica al servizio Amazon GameLift Servers che il processo del server è pronto per l'hosting delle sessioni di gioco. Chiama questo metodo dopo aver richiamato [InitSDK\(\)](#) e completato con successo le attività di configurazione necessarie prima che il processo del server possa ospitare una sessione di gioco. Questo metodo deve essere chiamato solo una volta per processo.

Sintassi

```
GenericOutcome ProcessReady(ProcessParameters processParameters)
```

Parameters

processParameters

Un oggetto [ProcessParameters](#) che comunica le informazioni seguenti sul processo del server:

- Nomi di metodi di callback, implementati nel codice del server di gioco, che il servizio Amazon GameLift Servers richiama per comunicare con il processo del server.
- Numero di porta sulla quale è in ascolto il processo del server.
- Percorso verso qualsiasi file specifico di una sessione di gioco che Amazon GameLift Servers deve acquisire e archiviare.

Obbligatorio: sì

Valore restituito

Restituisce un risultato generico costituito da successo o fallimento con un messaggio di errore.

Esempio

Questo esempio illustra la chiamata [ProcessReady\(\)](#) e le implementazioni della funzione delegata.

```
// Set parameters and call ProcessReady
var processParams = new ProcessParameters(
    this.OnGameSession,
    this.OnProcessTerminate,
    this.OnHealthCheck,
    this.OnGameSessionUpdate,
    port,
    new LogParameters(new List<string>()           // Examples of log and error files
        {
            "C:\\game\\logs",
            "C:\\game\\error"
        })
    written by the game server
);

var processReadyOutcome = GameLiftServerAPI.ProcessReady(processParams);

// Implement callback functions
void OnGameSession(GameSession gameSession)
{
    // game-specific tasks when starting a new game session, such as loading map
    // When ready to receive players
    var activateGameSessionOutcome = GameLiftServerAPI.ActivateGameSession();
}

void OnProcessTerminate()
{
    // game-specific tasks required to gracefully shut down a game session,
    // such as notifying players, preserving game state data, and other cleanup
    var ProcessEndingOutcome = GameLiftServerAPI.ProcessEnding();
}

bool OnHealthCheck()
{
    bool isHealthy;
    // complete health evaluation within 60 seconds and set health
```

```
    return isHealthy;  
}
```

RemovePlayerSession()

Notifica al servizio Amazon GameLift Servers che un giocatore con l'ID della sessione giocatore specificato si è disconnesso dal processo del server. In risposta, Amazon GameLift Servers modifica lo slot giocatore su disponibile, consentendone l'assegnazione a un nuovo giocatore.

Sintassi

```
GenericOutcome RemovePlayerSession(String playerId)
```

Parameters

playerSessionId

ID univoco rilasciato da Amazon GameLift Servers quando viene creata una nuova sessione di gioco. Un ID di sessione del giocatore viene specificato in un `PlayerSession` oggetto, che viene restituito in risposta a una chiamata del client alle azioni GameLift API [StartGameSessionPlacement](#), [CreateGameSession](#), [DescribeGameSessionPlacement](#), o [DescribePlayerSessions](#).

Tipo: stringa

Obbligatorio: sì

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

```
Aws::GameLift::GenericOutcome disconnectOutcome =  
    Aws::GameLift::Server::RemovePlayerSession(playerSessionId);
```

StartMatchBackfill()

Invia una richiesta per trovare nuovi giocatori per gli slot aperti in una sessione di gioco creata con FlexMatch. Vedi anche l'azione AWS SDK [StartMatchBackfill\(\)](#). Con questa operazione, è possibile

avviare le richieste di backfill degli abbinamenti da un processo del server di gioco che ospita la sessione di gioco. Scopri di più sulla funzione di [FlexMatchriempimento](#).

Questa operazione è asincrona. Se i nuovi giocatori sono stati abbinati in modo corretto, il servizio Amazon GameLift Servers invia i dati del matchmaker aggiornati tramite la funzione di callback `OnUpdateGameSession()`.

Un processo del server può avere un solo backfill degli abbinamenti attivo alla volta. Per inviare una nuova richiesta, chiama prima [StopMatchBackfill\(\)](#) per annullare la richiesta originale.

Sintassi

```
StartMatchBackfillOutcome StartMatchBackfill (StartMatchBackfillRequest  
startBackfillRequest);
```

Parameters

StartMatchBackfillRequest

Un oggetto [StartMatchBackfillRequest](#) che comunica le informazioni seguenti:

- ID del ticket da assegnare alla richiesta di backfill. Questa informazione è facoltativa; se l'ID non viene fornito, Amazon GameLift Servers ne creerà automaticamente uno.
- Matchmaker a cui inviare la richiesta. L'ARN di configurazione completo è obbligatorio. Questo valore può essere acquisito dai dati del matchmaker della sessione di gioco.
- ID della sessione di gioco che è in fase di backfilling.
- Dati di abbinamento disponibili per i giocatori correnti della sessione di gioco.

Obbligatorio: sì

Valore restituito

Restituisce un `StartMatchBackfillOutcome` oggetto con l'ID del ticket Match Backfill o restituisce un errore con un messaggio di errore.

Esempio

```
// Build a backfill request  
var startBackfillRequest = new AWS.GameLift.Server.Model.StartMatchBackfillRequest()  
{  
    TicketId = "a ticket ID", //optional
```



```

    MatchmakingConfigurationArn = "the matchmaker configuration ARN",
    GameSessionId = GameLiftServerAPI.GetGameSessionId().Result,    // gets ID for
current game session
    //get player data for all currently connected players
    MatchmakerData matchmakerData =
        MatchmakerData.FromJson(gameSession.MatchmakerData); // gets matchmaker
data for current players
    // get matchmakerData.Players
    // remove data for players who are no longer connected
    Players = ListOfPlayersRemainingInTheGame
};

// Send backfill request
var startBackfillOutcome = GameLiftServerAPI.StartMatchBackfill(startBackfillRequest);

// Implement callback function for backfill
void OnUpdateGameSession(GameSession myGameSession)
{
    // game-specific tasks to prepare for the newly matched players and update
    matchmaker data as needed
}

```

StopMatchBackfill()

Annulla una richiesta di backfill degli abbinamenti attiva creata con [StartMatchBackfill\(\)](#). Vedi anche l'azione AWS SDK [StopMatchmaking\(\)](#). Scopri di più sulla funzione di [FlexMatchriempimento](#).

Sintassi

```
GenericOutcome StopMatchBackfill (StopMatchBackfillRequest stopBackfillRequest);
```

Parameters

StopMatchBackfillRequest

Un oggetto [StopMatchBackfillRequest](#) che identifica il ticket di abbinamento da annullare:

- ID del ticket assegnato alla richiesta di backfill in fase di annullamento
- matchmaker a cui è stata inviata la richiesta di backfill
- sessione di gioco associata alla richiesta di backfill

Obbligatorio: sì

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

```
// Set backfill stop request parameters

var stopBackfillRequest = new AWS.GameLift.Server.Model.StopMatchBackfillRequest()
{
    TicketId = "a ticket ID", //optional, if not provided one is autogenerated
    MatchmakingConfigurationArn = "the matchmaker configuration ARN", //from the game
    session matchmaker data
    GameSessionId = GameLiftServerAPI.GetGameSessionId().Result //gets the ID for
    the current game session
};

var stopBackfillOutcome =
    GameLiftServerAPI.StopMatchBackfillRequest(stopBackfillRequest);
```

TerminateGameSession()

Questo metodo è obsoleto con la versione 4.0.1. Invece, il processo del server dovrebbe richiamare [ProcessEnding\(\)](#) dopo la fine di una sessione di gioco.

Notifica al Amazon GameLift Servers servizio che il processo del server ha terminato la sessione di gioco corrente. Questa azione viene richiamata quando il processo del server rimane attivo e pronto per ospitare una nuova sessione di gioco. Dovrebbe essere chiamata solo dopo il completamento della procedura di interruzione della sessione di gioco, perché segnala Amazon GameLift Servers che il processo del server è immediatamente disponibile per ospitare una nuova sessione di gioco.

Questa azione non viene eseguita se il processo del server verrà chiuso dopo l'interruzione della sessione di gioco. Invece, chiama [ProcessEnding\(\)](#) per segnalare che sia la sessione di gioco che il processo sul server stanno terminando.

Sintassi

```
GenericOutcome TerminateGameSession()
```

Parameters

Questa operazione non prevede parametri.

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

Esempio

Questo esempio illustra un processo del server al termine di una sessione di gioco.

```
// game-specific tasks required to gracefully shut down a game session,  
// such as notifying players, preserving game state data, and other cleanup  
  
var terminateGameSessionOutcome = GameLiftServerAPI.TerminateGameSession();  
var processReadyOutcome = GameLiftServerAPI.ProcessReady(processParams);
```

UpdatePlayerSessionCreationPolicy()

Aggiorna la capacità della sessione di gioco corrente di accettare nuove sessioni giocatore. Una sessione di gioco può essere configurata per accettare o rifiutare tutte le nuove sessioni giocatore. (Vedi anche l'azione [UpdateGameSession\(\)](#) nel Amazon GameLift ServersService API Reference).

Sintassi

```
GenericOutcome UpdatePlayerSessionCreationPolicy(PlayerSessionCreationPolicy  
    playerSessionPolicy)
```

Parameters

newPlayerSessionPolitica

Valore della stringa che indica se la sessione di gioco accetta nuovi giocatori.

Tipo: enum. [PlayerSessionCreationPolicy](#). I valori validi includono:

- ACCEPT_ALL: accetta tutte le nuove sessioni giocatore.
- DENY_ALL: rifiuta tutte le nuove sessioni giocatore.

Obbligatorio: sì

Valore restituito

Restituisce un risultato generico costituito da successo o fallimento con un messaggio di errore.

Esempio

Questo esempio definisce la policy di partecipazione alla sessione di gioco corrente per accettare tutti i giocatori.

```
var updatePlayerSessionCreationPolicyOutcome =  
  
    GameLiftServerAPI.UpdatePlayerSessionCreationPolicy(PlayerSessionCreationPolicy.ACCEPT_ALL);
```

Server SDK (Unreal) per Amazon GameLift Servers -- Azioni

Usa l'SDK del server per Unreal per integrare il tuo gioco multiplayer con cui ospitare. Amazon GameLift Servers Per indicazioni sul processo di integrazione, consulta [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#)

Note

Questo riferimento si riferisce a una versione precedente del server SDK for Amazon GameLift Servers. Per la versione più recente, consulta [Server C++ \(Unreal\) SDK 5.x per -- Azioni Amazon GameLift Servers](#).

Questa API è definita in `GameLiftServerSDK.h` e `GameLiftServerSDKModels.h`.

Per configurare il plugin Unreal Engine e vedere esempi di codice [Integrazione Amazon GameLift Servers in un progetto Unreal Engine](#).

Server SDK (Unreal) per Amazon GameLift Servers: tipi di dati

Utilizza il Amazon GameLift Servers server SDK for Unreal Reference per integrare il tuo gioco multiplayer con cui ospitarlo. Amazon GameLift Servers Per indicazioni sul processo di integrazione, consulta [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#)

Note

Questo riferimento si riferisce a una versione precedente del server SDK for Amazon GameLift Servers. Per la versione più recente, consulta [Server C++ \(Unreal\) SDK 5.x per Amazon GameLift Servers -- Tipi di dati](#).

Questa API è definita in `GameLiftServerSDK.h` e `GameLiftServerSDKModels.h`.

Per configurare il plugin Unreal Engine e vedere esempi di codice [Integrazione Amazon GameLift Servers in un progetto Unreal Engine](#).

[Server SDK \(Unreal\) per Amazon GameLift Servers -- Azioni](#)

Argomenti

- [FDescribePlayerSessionsRequest](#)
- [FProcessParametri](#)
- [FStartMatchBackfillRequest](#)
- [FStopMatchBackfillRequest](#)

FDescribePlayerSessionsRequest

Questo tipo di dati viene utilizzato per specificare quale sessione del giocatore recuperare. Puoi utilizzarlo come segue:

- Fornisci un `PlayerSessionId` messaggio per richiedere una sessione di gioco specifica.
- Fornisci un `GameSessionId` campo per richiedere tutte le sessioni dei giocatori nella sessione di gioco specificata.
- Fornisci un `PlayerId` campo per richiedere tutte le sessioni di gioco per il giocatore specificato.

Per le raccolte di grandi dimensioni delle sessioni giocatore, utilizza i parametri di paginazione per recuperare i risultati in blocchi sequenziali.

Indice

GameSessionId

Identificatore univoco della sessione di gioco. Utilizzare questo parametro per richiedere tutte le sessioni giocatore per la sessione di gioco specificata. Il formato dell'ID della sessione di gioco è il seguente: `arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string>`. Il valore di `<ID string>` corrisponde a una stringa ID personalizzata o (se ne è stata specificata una al momento della creazione della sessione di gioco) a una stringa generata.

─Tipo: stringa

Obbligatorio: no

Limite

Numero massimo di risultati da restituire. Utilizzate questo parametro con NextToken per ottenere risultati sotto forma di set di pagine sequenziali. Se viene specificato un ID sessione giocatore, questo parametro verrà ignorato.

Tipo: Integer

Obbligatorio: no

NextToken

Token che indica l'inizio della pagina sequenziale successiva relativa ai risultati. Utilizzare il token restituito con una chiamata precedente a questa operazione. Per specificare l'inizio del set di risultati, non specificare un valore. Se viene specificato un ID sessione giocatore, questo parametro verrà ignorato.

─Tipo: stringa

Obbligatorio: no

PlayerId

Identificatore univoco per un giocatore. IDs I giocatori sono definiti dallo sviluppatore. Per informazioni, consulta [Genera giocatore IDs](#).

─Tipo: stringa

Obbligatorio: no

PlayerSessionId

Identificatore univoco della sessione giocatore.

─Tipo: stringa

Obbligatorio: no

PlayerSessionStatusFilter

Stato sessione giocatore su cui filtrare i risultati. Tra gli stati sessione giocatore possibili sono inclusi i seguenti:

- **RISERVATO** — La richiesta di sessione del giocatore è stata ricevuta, ma il giocatore non si è ancora connesso al server. Il processo and/or è stato convalidato.

- **ACTIVE (ATTIVO)** - Il giocatore è stato convalidato dal processo del server ed è attualmente collegato.
- **COMPLETED (COMPLETATO)** - La connessione del giocatore è stata interrotta.
- **TIMEDOUT** — È stata ricevuta una richiesta di sessione di gioco, ma il giocatore che non si and/or è connesso non è stata convalidata entro il limite di timeout (60 secondi).

─Tipo: stringa

Obbligatorio: no

FProcessParametri

Questo tipo di dati contiene il set di parametri inviato al servizio Amazon GameLift Servers in una chiamata [ProcessReady\(\)](#).

Indice

port

Numero di porta a cui il processo del server rimarrà in attesa di nuove connessioni dei giocatori. Il valore deve rientrare nella gamma di porte configurate per qualsiasi parco istanze che distribuisce questa build del server di gioco. Questo numero di porta è incluso nella sessione di gioco e negli oggetti della sessione del giocatore utilizzati dalle sessioni di gioco per la connessione a un processo server.

Tipo: numero intero

Obbligatorio: sì

logParameters

Un oggetto con un elenco di percorsi delle directory per i file log delle sessioni di gioco.

Tipo: TArray < FString >

Obbligatorio: no

onStartGameSessione

Nome della funzione di callback richiamata dal Amazon GameLift Servers servizio per attivare una nuova sessione di gioco. Amazon GameLift Servers chiama questa funzione in risposta alla richiesta del client. [CreateGameSession](#) La funzione di callback accetta un [GameSession](#) oggetto (definito nel Amazon GameLift ServersService API Reference).

Tipo: FOnStartGameSession

Obbligatorio: sì

onProcessTerminate

Nome della funzione di callback richiamata dal Amazon GameLift Servers servizio per forzare la chiusura del processo del server. Dopo aver richiamato questa funzione, Amazon GameLift Servers attende per cinque minuti l'arresto del processo del server e risponde con una chiamata [ProcessEnding\(\)](#) prima di arrestare il processo del server.

Tipo: Delegato FSimple

Obbligatorio: no

onHealthCheck

Nome della funzione di callback richiamata dal Amazon GameLift Servers servizio per richiedere un rapporto sullo stato di salute del processo del server. Amazon GameLift Servers chiama questa funzione ogni 60 secondi. Dopo aver richiamato questa funzione, Amazon GameLift Servers attende per 60 secondi l'arrivo di una risposta e, se non ne riceve alcuna, registra il processo del server come non integro.

Tipo: FOnHealthCheck

Obbligatorio: no

onUpdateGameSessione

Nome della funzione di callback richiamata dal Amazon GameLift Servers servizio per passare un oggetto di sessione di gioco aggiornato al processo del server. Amazon GameLift Servers chiama questa funzione quando è stata elaborata una richiesta di [match backfill](#) per fornire dati aggiornati sul matchmaker. Passa un [GameSession](#) oggetto, uno status update (updateReason) e l'ID del ticket match backfill.

Tipo: FOnUpdateGameSession

Obbligatorio: no

FStartMatchBackfillRequest

Questo tipo di dati viene utilizzato per inviare una richiesta di backfill di abbinamento. Le informazioni vengono comunicate al servizio Amazon GameLift Servers in una chiamata [StartMatchBackfill\(\)](#).

Indice

GameSessionArn

Identificatore univoco della sessione di gioco. L'operazione API [GetGameSessionId\(\)](#) restituisce l'identificatore nel formato ARN.

Tipo: FString

Obbligatorio: sì

MatchmakingConfigurationArn

Identificatore univoco, sotto forma di un ARN, che il matchmaker utilizza per questa richiesta. Per trovare il matchmaker utilizzato per creare la sessione di gioco originale, esaminare l'oggetto sessione di gioco nella proprietà dei dati del matchmaker. Scopri di più sui dati dei matchmaker in [Lavora con i dati dei matchmaker](#).

Tipo: FString

Obbligatorio: sì

Players

Un set di dati che rappresenta tutti i giocatori che sono attualmente impegnati nella sessione di gioco. Il matchmaker utilizza queste informazioni per cercare nuovi giocatori che rappresentino un buon abbinamento per i giocatori attuali. Consulta la Guida di riferimento Amazon GameLift Servers API per una descrizione del formato degli oggetti Player. Per trovare gli attributi dei giocatori e gli incarichi della squadra, cerca nell'oggetto della sessione di gioco, nella proprietà dei dati del matchmaker. IDs Se la latenza viene utilizzata dal matchmaker, raccogliere la latenza aggiornata per la regione attuale e includerla nei dati di ciascun giocatore.

[Digita: < > TArray FPlayer](#)

Obbligatorio: sì

TicketId

Identificatore univoco per un abbinamento o un ticket di richiesta di backfill degli abbinamenti. Se non viene fornito alcun valore qui, ne Amazon GameLift Servers genererà uno sotto forma di UUID. Utilizzare questo identificatore per monitorare lo stato del ticket di backfill degli abbinamenti o annullare la richiesta, se necessario.

Tipo: FString

Obbligatorio: no

FStopMatchBackfillRequest

Questo tipo di dati viene utilizzato per annullare una richiesta di backfill di abbinamento. Le informazioni vengono comunicate al servizio Amazon GameLift Servers in una chiamata [StopMatchBackfill\(\)](#).

Indice

GameSessionArn

Identificatore univoco della sessione di gioco associato alla richiesta in fase di annullamento.

Tipo: FString

Obbligatorio: sì

MatchmakingConfigurationArn

Identificatore univoco del matchmaker a cui è stata inviata questa richiesta.

Tipo: FString

Obbligatorio: sì

TicketId

Identificatore univoco del ticket di richiesta di backfill da annullare.

Tipo: FString

Obbligatorio: sì

[Server SDK \(Unreal\) per Amazon GameLift Servers: tipi di dati](#)

Argomenti

- [AcceptPlayerSession\(\)](#)
- [ActivateGameSession\(\)](#)

- [DescribePlayerSessions\(\)](#)
- [GetGameSessionId\(\)](#)
- [GetInstanceCertificate\(\)](#)
- [GetSdkVersion\(\)](#)
- [InitSDK\(\)](#)
- [ProcessEnding\(\)](#)
- [ProcessReady\(\)](#)
- [RemovePlayerSession\(\)](#)
- [StartMatchBackfill\(\)](#)
- [StopMatchBackfill\(\)](#)
- [TerminateGameSession\(\)](#)
- [UpdatePlayerSessionCreationPolicy\(\)](#)

AcceptPlayerSession()

Notifica al Amazon GameLift Servers servizio che un giocatore con l'ID di sessione del giocatore specificato si è connesso al processo del server e deve essere convalidato. Amazon GameLift Servers verifica che l'ID della sessione del giocatore sia valido, ovvero che l'ID giocatore abbia riservato uno slot nella sessione di gioco. Dopo la convalida, Amazon GameLift Servers modifica lo stato dello slot giocatore da RESERVED (RISERVATO) ad ACTIVE (ATTIVO).

Sintassi

```
FGameLiftGenericOutcome AcceptPlayerSession(const FString& playerId)
```

Parameters

playerSessionId

ID univoco rilasciato dal Amazon GameLift Servers servizio in risposta a una chiamata all'azione dell'API AWS SDK Amazon GameLift Servers. [CreatePlayerSession](#) Il client di gioco fa riferimento a questo ID durante la connessione al processo del server.

Tipo: FString

Obbligatorio: sì

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

ActivateGameSession()

Notifica al servizio Amazon GameLift Servers che il processo del server ha attivato una sessione di gioco ed è pronto per ricevere le connessioni dei giocatori. Questa operazione deve essere chiamata come parte della funzione di callback `onStartGameSession()`, dopo il completamento dell'inizializzazione di tutte le sessioni di gioco.

Sintassi

```
FGameLiftGenericOutcome ActivateGameSession()
```

Parameters

Questa operazione non prevede parametri.

Valore restituito

Restituisce un risultato generico costituito da esito positivo o negativo con un messaggio di errore.

DescribePlayerSessions()

Recupera i dati della sessione giocatore, tra cui le impostazioni, i metadati della sessione e i dati dei giocatori. Utilizza questa operazione per ottenere le informazioni per una singola sessione giocatore, per tutte le sessioni giocatore in una sessione di gioco o per tutte le sessioni giocatore associate a un singolo ID giocatore.

Sintassi

```
FGameLiftDescribePlayerSessionsOutcome DescribePlayerSessions(const  
FGameLiftDescribePlayerSessionsRequest &describePlayerSessionsRequest)
```

Parameters

describePlayerSessionsRichiesta

Un oggetto [FDescribePlayerSessionsRequest](#) che descrive le sessioni giocatore da recuperare.

Obbligatorio: sì

Valore restituito

Se l'esito è positivo, restituisce un oggetto [FDescribePlayerSessionsRequest](#) contenente un set di oggetti di sessione giocatore corrispondente ai parametri della richiesta. Gli oggetti della sessione Player hanno una struttura identica al tipo di [PlayerSession](#) dati dell'Amazon GameLift ServersAPI AWS SDK.

GetGameSessionId()

Recupera l'ID della sessione di gioco attualmente ospitata dal processo del server, se il processo del server è attivo.

Sintassi

```
FGameLiftStringOutcome GetGameSessionId()
```

Parameters

Questa operazione non prevede parametri.

Valore restituito

Se l'esito è positivo, l'ID della sessione di gioco verrà restituito come oggetto `FGameLiftStringOutcome`. Se l'esito è negativo, verrà restituito un messaggio di errore.

GetInstanceCertificate()

Recupera la posizione del file di un certificato TLS con codifica pem associato alla flotta e alle relative istanze. AWS Certificate Manager genera questo certificato quando crei un nuovo parco veicoli con la configurazione del certificato impostata su GENERATED. Utilizza questo certificato per stabilire una connessione sicura con un client di gioco e per crittografare la comunicazione client/server.

Sintassi

```
FGameLiftGetInstanceCertificateOutcome GetInstanceCertificate()
```

Parameters

Questa operazione non prevede parametri.

Valore restituito

In caso di successo, restituisce un `GetInstanceCertificateOutcome` oggetto contenente la posizione del file di certificato TLS e della catena di certificati della flotta, che sono archiviati nell'istanza. Nell'istanza viene inoltre archiviato un file di certificato radice, estratto dalla catena di certificati. Se l'esito è negativo, verrà restituito un messaggio di errore.

Per ulteriori informazioni sul certificato e sui dati della catena di certificati, consulta [GetCertificate Response Elements](#) nell' AWS Certificate Manager API Reference.

GetSdkVersion()

Restituisce il numero di versione corrente dell'SDK integrato nel processo del server.

Sintassi

```
FGameLiftStringOutcome GetSdkVersion();
```

Parameters

Questa operazione non prevede parametri.

Valore restituito

Se l'esito è positivo, restituisce la versione corrente dell'SDK come oggetto `FGameLiftStringOutcome`. La stringa restituita include solo il numero di versione (ad esempio «3.1.5»). Se l'esito è negativo, verrà restituito un messaggio di errore.

Esempio

```
Aws::GameLift::AwsStringOutcome SdkVersionOutcome =  
    Aws::GameLift::Server::GetSdkVersion();
```

InitSDK()

Inizializza l'SDK Amazon GameLift Servers. Questo metodo deve essere chiamato all'avvio, prima che avvenga qualsiasi altra inizializzazione correlata a Amazon GameLift Servers.

Sintassi

```
FGameLiftGenericOutcome InitSDK()
```

Parameters

Questa operazione non prevede parametri.

Valore restituito

Restituisce un risultato generico costituito da esito positivo o negativo con un messaggio di errore.

ProcessEnding()

Avvisa il servizio Amazon GameLift Servers che il processo del server è in fase di arresto. Questo metodo deve essere richiamato dopo tutte le altre attività di pulizia, tra cui l'arresto di tutte le sessioni di gioco attive. Questo metodo deve uscire con il codice 0; un codice di uscita diverso da zero genera un messaggio di evento che indica che il processo non è terminato correttamente.

Sintassi

```
FGameLiftGenericOutcome ProcessEnding()
```

Parameters

Questa operazione non prevede parametri.

Valore restituito

Restituisce un risultato generico costituito da esito positivo o negativo con un messaggio di errore.

ProcessReady()

Notifica al servizio Amazon GameLift Servers che il processo del server è pronto per l'hosting delle sessioni di gioco. Chiama questo metodo dopo aver richiamato [InitSDK\(\)](#) e completato con successo le attività di configurazione necessarie prima che il processo del server possa ospitare una sessione di gioco. Questo metodo deve essere chiamato solo una volta per processo.

Sintassi

```
FGameLiftGenericOutcome ProcessReady(FProcessParameters &processParameters)
```

Parameters

FProcessParametri

Un oggetto [FProcessParametri](#) che comunica le informazioni seguenti sul processo del server:

- Nomi di metodi di callback, implementati nel codice del server di gioco, che il servizio Amazon GameLift Servers richiama per comunicare con il processo del server.
- Numero di porta sulla quale è in ascolto il processo del server.
- Percorso verso qualsiasi file specifico di una sessione di gioco che Amazon GameLift Servers deve acquisire e archiviare.

Obbligatorio: sì

Valore restituito

Restituisce un risultato generico costituito da esito positivo o negativo con un messaggio di errore.

Esempio

Consulta il codice di esempio nella sezione relativa all'[utilizzo del plug-in Unreal Engine](#).

RemovePlayerSession()

Notifica al servizio Amazon GameLift Servers che un giocatore con l'ID della sessione giocatore specificato si è disconnesso dal processo del server. In risposta, Amazon GameLift Servers modifica lo slot giocatore su disponibile, consentendone l'assegnazione a un nuovo giocatore.

Sintassi

```
FGameLiftGenericOutcome RemovePlayerSession(const FString& playerId)
```

Parameters

playerSessionId

ID univoco rilasciato dal Amazon GameLift Servers servizio in risposta a una chiamata all'azione [CreatePlayerSession](#) dell'Amazon GameLift Servers API AWS SDK. Il client di gioco fa riferimento a questo ID durante la connessione al processo del server.

Tipo: FString

Obbligatorio: sì

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

StartMatchBackfill()

Invia una richiesta per trovare nuovi giocatori per gli slot aperti in una sessione di gioco creata con FlexMatch. Vedi anche l'azione AWS SDK [StartMatchBackfill\(\)](#). Con questa operazione, è possibile avviare le richieste di backfill degli abbinamenti da un processo del server di gioco che ospita la sessione di gioco. Scopri di più sulla funzione di [FlexMatchriempimento](#).

Questa operazione è asincrona. Se i nuovi giocatori sono stati abbinati in modo corretto, il servizio Amazon GameLift Servers invia i dati del matchmaker aggiornati tramite la funzione di callback `OnUpdateGameSession()`.

Un processo del server può avere un solo backfill degli abbinamenti attivo alla volta. Per inviare una nuova richiesta, chiama prima [StopMatchBackfill\(\)](#) per annullare la richiesta originale.

Sintassi

```
FGameLiftStringOutcome StartMatchBackfill (FStartMatchBackfillRequest  
&startBackfillRequest);
```

Parameters

FStartMatchBackfillRequest

Un oggetto [FStartMatchBackfillRequest](#) che comunica le informazioni seguenti:

- ID del ticket da assegnare alla richiesta di backfill. Questa informazione è facoltativa; se l'ID non viene fornito, Amazon GameLift Servers ne creerà automaticamente uno.
- Matchmaker a cui inviare la richiesta. L'ARN di configurazione completo è obbligatorio. Questo valore può essere acquisito dai dati del matchmaker della sessione di gioco.
- ID della sessione di gioco che è in fase di backfilling.
- Dati di abbinamento disponibili per i giocatori correnti della sessione di gioco.

Obbligatorio: sì

Valore restituito

Se completato, restituisce il ticket di backfill degli abbinamenti come oggetto `FGameLiftStringOutcome`. Se l'esito è negativo, verrà restituito un messaggio di errore. [Lo stato del ticket può essere monitorato utilizzando l'azione AWS DescribeMatchmaking SDK \(\)](#).

StopMatchBackfill()

Annulla una richiesta di backfill degli abbinamenti attiva creata con [StartMatchBackfill\(\)](#). [Vedi anche l'azione AWS StopMatchmaking SDK \(\)](#). Scopri di più sulla funzione di [FlexMatchriempimento](#).

Sintassi

```
FGameLiftGenericOutcome StopMatchBackfill (FStopMatchBackfillRequest  
&stopBackfillRequest);
```

Parameters

StopMatchBackfillRequest

Un oggetto [FStopMatchBackfillRequest](#) che identifica il ticket di abbinamento da annullare:

- ID del ticket assegnato alla richiesta di backfill in fase di annullamento
- matchmaker a cui è stata inviata la richiesta di backfill
- sessione di gioco associata alla richiesta di backfill

Obbligatorio: sì

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

TerminateGameSession()

Questo metodo è obsoleto con la versione 4.0.1. Invece, il processo del server dovrebbe richiamare [ProcessEnding\(\)](#) dopo la fine di una sessione di gioco.

Notifica al Amazon GameLift Servers servizio che il processo del server ha terminato la sessione di gioco corrente. Questa azione viene richiamata quando il processo del server rimane attivo e pronto per ospitare una nuova sessione di gioco. Dovrebbe essere chiamata solo dopo il completamento della procedura di interruzione della sessione di gioco, perché segnala Amazon GameLift Servers che il processo del server è immediatamente disponibile per ospitare una nuova sessione di gioco.

Questa azione non viene eseguita se il processo del server verrà chiuso dopo l'interruzione della sessione di gioco. Invece, chiama [ProcessEnding\(\)](#) per segnalare che sia la sessione di gioco che il processo sul server stanno terminando.

Sintassi

```
FGameLiftGenericOutcome TerminateGameSession()
```

Parameters

Questa operazione non prevede parametri.

Valore restituito

Restituisce un risultato generico composto da successo o fallimento con un messaggio di errore.

UpdatePlayerSessionCreationPolicy()

Aggiorna la capacità della sessione di gioco corrente di accettare nuove sessioni giocatore. Una sessione di gioco può essere configurata per accettare o rifiutare tutte le nuove sessioni giocatore. (Vedi anche l'operazione [UpdateGameSession\(\)](#) nella documentazione di riferimento per l'API del servizio Amazon GameLift Servers.)

Sintassi

```
FGameLiftGenericOutcome UpdatePlayerSessionCreationPolicy(EPlayerSessionCreationPolicy policy)
```

Parameters

Policy

Valore che indica se la sessione di gioco accetta nuovi giocatori.

Tipo: enum. `EPlayerSessionCreationPolicy`. I valori validi includono:

- `ACCEPT_ALL`: accetta tutte le nuove sessioni giocatore.
- `DENY_ALL`: rifiuta tutte le nuove sessioni giocatore.

Obbligatorio: sì

Valore restituito

Restituisce un risultato generico costituito da esito positivo o negativo con un messaggio di errore.

Eventi di collocamento delle sessioni di gioco

Amazon GameLift Serversemette eventi per ogni richiesta di posizionamento della sessione di gioco man mano che viene elaborata. Puoi pubblicare questi eventi su un argomento di Amazon SNS, come descritto in. [Imposta la notifica degli eventi per il posizionamento della sessione di gioco](#) Questi eventi vengono inoltre trasmessi ad Amazon CloudWatch Events quasi in tempo reale e con la massima diligenza possibile.

Questo argomento descrive la struttura degli eventi di collocamento delle sessioni di gioco e fornisce un esempio per ogni tipo di evento. Per ulteriori informazioni sullo stato delle richieste di posizionamento delle sessioni di gioco, consulta [GameSessionPlacement](#)l'Amazon GameLift ServersAPI Reference.

Sintassi degli eventi di posizionamento

Gli eventi sono rappresentati come oggetti JSON. La struttura degli eventi è conforme al modello CloudWatch Events, con campi di primo livello simili e dettagli specifici del servizio.

I campi di primo livello includono quanto segue (vedi [Event Pattern](#) per maggiori dettagli):

version

Questo campo è sempre impostato su 0 (zero).

id

Identificatore di tracciamento univoco per l'evento.

detail-type (tipo di dettaglio)

Il valore è sempreGameLift Queue Placement Event.

origine

Il valore è sempreaws .gamelift.

account

L' AWS account che viene utilizzato per gestireAmazon GameLift Servers.

time

Timestamp dell'evento.

region

La AWS regione in cui viene elaborata la richiesta di collocamento. Questa è la regione in cui si trova la coda della sessione di gioco in uso.

risorse

Valore ARN della coda della sessione di gioco che sta elaborando la richiesta di posizionamento.

PlacementFulfilled

La richiesta di collocamento è stata soddisfatta con successo. È stata avviata una nuova sessione di gioco e sono state create nuove sessioni per ogni giocatore elencato nella richiesta di posizionamento della sessione di gioco. Le informazioni sulla connessione del giocatore sono disponibili.

Sintassi dettagliata:

PlacementID

Un identificatore univoco assegnato alla richiesta di posizionamento della sessione di gioco.

port

Il numero di porta per la nuova sessione di gioco.

gameSessionArn

L'identificatore ARN per la nuova sessione di gioco.

ipAddress

L'indirizzo IP della sessione di gioco.

Nome DNS

L'identificatore DNS assegnato all'istanza che esegue la nuova sessione di gioco. Il formato dei valori è diverso a seconda che l'istanza che esegue la sessione di gioco sia abilitata per TLS. Quando si connettono a una sessione di gioco su una flotta che supporta TLS, i giocatori devono utilizzare il nome DNS, non l'indirizzo IP.

Flotte abilitate per TLS: `<unique identifier>.<region identifier>.amazongamelift.com`

Non-TLS-enabled flotte: `ec2-<unique identifier>.compute.amazonaws.com`

`startTime`

Timestamp che indica quando questa richiesta è stata inserita nella coda.

`endTime`

Timestamp che indica quando questa richiesta è stata soddisfatta.

`gameSessionRegion`

AWS Regione di residenza della flotta che ospita la sessione di gioco.

`gameSessionLocation`

L'ubicazione della flotta in cui è in corso la sessione di gioco. Questo valore potrebbe specificare la regione di origine della flotta o una località remota.

`playerGatewayStatus`

Lo stato del gateway del giocatore presso la sede della flotta che esegue questa sessione di gioco.

I valori possibili includono:

- **ENABLED**— Il Player Gateway è disponibile per instradare le connessioni dei giocatori per questa sessione di gioco.
- **DISABLED**— Il gateway del giocatore non è disponibile per questa sessione di gioco.

`ComputeName`

Un'etichetta descrittiva per la risorsa di calcolo che ospita la sessione di gioco. Per le flotte EC2, questo è l'ID dell'istanza EC2. Per le flotte di container, a ciascun gruppo di container di server di gioco su un'istanza di flotta viene assegnato un nome di calcolo. Per le flotte Anywhere, questo è il nome di calcolo personalizzato.

`placedPlayerSessions`

La raccolta di sessioni di gioco che sono state create per ogni giocatore nella richiesta di posizionamento della sessione di gioco.

Esempio

```
{  
  "version": "0",  
}
```

```

{id": "1111aaaa-bb22-cc33-dd44-5555eeee66ff",
"detail-type": "GameLift Queue Placement Event",
"source": "aws.gamelift",
"account": "123456789012",
"time": "2021-03-01T15:50:52Z",
"region": "us-east-1",
"resources": [
  "arn:aws:gamelift:us-west-2:123456789012:gamesessionqueue/MegaFrogRace-NA"
],
"detail": {
  "type": "PlacementFulfilled",
  "placementId": "9999ffff-88ee-77dd-66cc-5555bb44aa",
  "port": "6262",
  "gameSessionArn": "arn:aws:gamelift:us-west-2::gamesession/
fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa/us-
east-1/4444dddd-55ee-66ff-77aa-8888bbbb99cc",
  "ipAddress": "98.987.98.987",
  "dnsName": "ec2-12-345-67-890.us-west-2.compute.amazonaws.com",
  "startTime": "2021-03-01T15:50:49.741Z",
  "endTime": "2021-03-01T15:50:52.084Z",
  "gameSessionRegion": "us-west-2",
  "playerGatewayStatus": "ENABLED",
  "gameSessionLocation": "us-east-1",
  "computeName": "i-1234567890abcdef0",
  "placedPlayerSessions": [
    {
      "playerId": "player-1"
      "playerSessionId": "psess-1232131232324124123123"
    }
  ]
}
}

```

PlacementCancelled

La richiesta di collocamento è stata annullata con una chiamata al GameLift servizio.

[StopGameSessionPlacement](#)

Dettaglio:

ID di posizionamento

Un identificatore univoco assegnato alla richiesta di posizionamento della sessione di gioco.

startTime

Timestamp che indica quando questa richiesta è stata inserita nella coda.

endTime

Timestamp che indica quando questa richiesta è stata annullata.

Esempio

```
{
  "version": "0",
  "id": "1111aaaa-bb22-cc33-dd44-5555eeee66ff",
  "detail-type": "GameLift Queue Placement Event",
  "source": "aws.gamelift",
  "account": "123456789012",
  "time": "2021-03-01T15:50:52Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:gamelift:us-west-2:123456789012:gamesessionqueue/MegaFrogRace-NA"
  ],
  "detail": {
    "type": "PlacementCancelled",
    "placementId": "9999ffff-88ee-77dd-66cc-5555bb44aa",
    "startTime": "2021-03-01T15:50:49.741Z",
    "endTime": "2021-03-01T15:50:52.084Z"
  }
}
```

PlacementTimedOut

Il posizionamento della sessione di gioco non è stato completato correttamente prima della scadenza del limite di tempo della coda. La richiesta di piazzamento può essere inoltrata nuovamente se necessario.

Dettaglio:

ID di posizionamento

Un identificatore univoco assegnato alla richiesta di posizionamento della sessione di gioco.

startTime

Timestamp che indica quando questa richiesta è stata inserita nella coda.

endTime

Timestamp che indica quando questa richiesta è stata annullata.

Esempio

```
{
  "version": "0",
  "id": "1111aaaa-bb22-cc33-dd44-5555eeee66ff",
  "detail-type": "GameLift Queue Placement Event",
  "source": "aws.gamelift",
  "account": "123456789012",
  "time": "2021-03-01T15:50:52Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:gamelift:us-west-2:123456789012:gamesessionqueue/MegaFrogRace-NA"
  ],
  "detail": {
    "type": "PlacementTimedOut",
    "placementId": "9999ffff-88ee-77dd-66cc-5555bb44aa",
    "startTime": "2021-03-01T15:50:49.741Z",
    "endTime": "2021-03-01T15:50:52.084Z"
  }
}
```

PlacementFailed

Amazon GameLift Servers non è stato in grado di soddisfare la richiesta di sessione di gioco. Ciò è generalmente causato da un errore interno imprevisto. La richiesta di collocamento può essere inoltrata nuovamente se necessario.

Dettaglio:

ID di posizionamento

Un identificatore univoco assegnato alla richiesta di posizionamento della sessione di gioco.

startTime

Timestamp che indica quando questa richiesta è stata inserita nella coda.

endTime

Timestamp che indica quando questa richiesta non è riuscita.

Esempio

```
{
  "version": "0",
  "id": "39c978f3-ba46-3f7c-e787-55bfcca1bd31",
  "detail-type": "GameLift Queue Placement Event",
  "source": "aws.gamelift",
  "account": "252386620677",
  "time": "2021-03-01T15:50:52Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:gamelift:us-west-2:252386620677:gamesessionqueue/MegaFrogRace-NA"
  ],
  "detail": {
    "type": "PlacementFailed",
    "placementId": "e4a1119a-39af-45cf-a990-ef150fe0d453",
    "startTime": "2021-03-01T15:50:49.741Z",
    "endTime": "2021-03-01T15:50:52.084Z"
  }
}
```

Amazon GameLift ServersVersioni AMI

La tabella seguente identifica le ultime immagini di macchine Amazon (AMIs) Amazon GameLift Servers utilizzate per l'hosting EC2 gestito. Come descritto in [Analisi della configurazione e delle vulnerabilità in Amazon GameLift Servers](#), è necessario creare regolarmente nuove flotte EC2 Amazon GameLift Servers gestite per distribuire gli ultimi aggiornamenti delle versioni AMI.

AMIs da utilizzare con server SDK for 5+ Amazon GameLift Servers

Amazon GameLift Servers utilizza quanto segue AMIs per ospitare server di gioco integrati con il server SDK per Amazon GameLift Servers la versione 5.

Amazon GameLift Servers immagine	Architettura	Patch più recente	AWS Immagine base
Amazon Linux 2023 BASE_AMI_LINUX_2023	x86	2026-03-27	Amazon Linux 2023 AMI 2023.10.2 0260302.1 x86_64 HVM kernel-6.1 (note di rilascio)
Amazon Linux 2 BASE_AMI_LINUX_2_SDK_5	x86	2026-03-27	Amazon Linux 2 Kernel 5.10 AMI 2.0.20260302.0 x86_64 HVM gp2 (note di rilascio)
Windows 2016 BASE_AMI_WINDOWS_2016_SDK_5	x86	2026-03-27	Windows_Server-2016-Inglese-Full-Base-2026.03.11 (note di rilascio)
Amazon Linux 2023 BASE_AMI_LINUX_2023_ARM	ARM64	2026-03-27	Amazon Linux 2023 AMI 2023.10.2 0260302.1 arm64 HVM kernel-6.1 (note di rilascio)
Amazon Linux 2 BASE_AMI_LINUX_2_ARM	ARM64	2026-03-27	Amazon Linux 2 LTS Arm64 Kernel 5.10 AMI 2.0.20260302.0 arm64 HVM gp2 (note di rilascio)
Windows 2022 BASE_AMI_WINDOWS_2022	x86	2026-03-27	Windows_Server-2022-Inglese-Full-Base-2026.03.11 (note di rilascio)

AMIs da utilizzare con server SDK for 4 Amazon GameLift Servers

Amazon GameLift Servers utilizza quanto segue AMIs per ospitare server di gioco integrati con server SDK per la Amazon GameLift Servers versione 4 o precedente.

Amazon GameLift Servers immagine	Architettura	Patch più recente	AWS Immagine base
Amazon Linux 2 BASE_AMI_LINUX_2_SDK_4	x86	2026-03-27	AMI Amazon Linux 2 2.0.20260302.0 x86_64 HVM gp2 (note di rilascio)
Amazon Linux 2023 BASE_AMI_LINUX_2023_SDK_4 Disponibile solo nelle regioni Cina (Pechino) e Cina (Ningxia).	x86	2026-03-27	Amazon Linux 2023 AMI 2023.10.20260302.1 x86_64 HVM kernel-6.1 (note di rilascio)
Windows 2016 BASE_AMI_WINDOWS_2016_SDK_4	x86	2026-03-27	Windows_Server-2016-Inglese-Full-Base-2026.03.11 (note di rilascio)

Per maggiori informazioni, consulta le seguenti risorse:

- [Note di rilascio di Amazon Linux 2023](#)
- [Note di rilascio di Amazon Linux 2](#)
- [AWS Cronologia delle versioni di Windows AMI](#)
- [Descrizione delle modifiche ai contenuti di Software Update Services e Windows Server Update Services per il 2024](#)

Amazon GameLift Servers sedi di assistenza

Amazon GameLift Servers le funzionalità sono disponibili su più Regioni AWS Local Zones. Puoi progettare una soluzione di hosting che colloca i server di gioco esattamente dove si trovano i giocatori.

AWS Località supportate

La tabella seguente riporta un elenco delle Regioni AWS Local Zones che supportano Amazon GameLift Servers le risorse. Indica i tipi di risorse che è possibile creare in ogni posizione.

Posizione geografica	Codice di ubicazione	Regione d'origine per le flotte gestite (sede unica)	Area geografica per le flotte gestite (con più sedi)	Localizzazione remota per flotte gestite (più sedi)	Flotta ovunque	Coda della sessione di gioco	FlexMatchmaking e set di regole	DDoProtection S (player gateway)
Stati Uniti orientali (Virginia settentrionale)	us-east-1	Sì	Sì	Sì	Sì	Sì	Sì	Sì [†]
Stati Uniti orientali (Ohio)	us-east-2	Sì		Sì	Sì	Sì		
Stati Uniti occidentali (California settentrionale)	us-west-1	Sì		Sì	Sì	Sì		
Stati Uniti occidentali (Oregon)	us-west-2	Sì	Sì	Sì	Sì	Sì	Sì	Sì [†]
Africa (Città del Capo)	af-south-1			Sì				

Posizione geografica	Codice di ubicazione	Regione d'origine per le flotte gestite (sede unica)	Area geografica per le flotte gestite (con più sedi)	Localizzazione remota per flotte gestite (più sedi)	Flotta ovunque	Coda della sessione di gioco	FlexMatchmaker e set di regole	DDoProtezione S (player gateway)
Asia Pacifico (Thailandia)	ap-southeast-7			Sì				
Asia Pacifico (Hong Kong)	ap-east-1			Sì				
Asia Pacifico (Malesia)	ap-southeast-5			Sì				
Asia Pacifico (Mumbai)	ap-south-1	Sì		Sì	Sì	Sì		
Asia Pacifico (Osaka)	ap-northeast-3			Sì				
Asia Pacifico (Seoul)	ap-northeast-2	Sì	Sì	Sì	Sì	Sì	Sì	Sì [†]
Asia Pacifico (Singapore)	ap-southeast-1	Sì		Sì	Sì	Sì		
Asia Pacifico (Sydney)	ap-southeast-2	Sì	Sì	Sì	Sì	Sì	Sì	Sì [†]

Posizione geografica	Codice di ubicazione	Regione d'origine per le flotte gestite (sede unica)	Area geografica per le flotte gestite (con più sedi)	Localizzazione remota per flotte gestite (più sedi)	Flotta ovunque	Coda della sessione di gioco	FlexMatchmaker e set di regole	DDoProtection S (player gateway)
Asia Pacifico (Tokyo)	ap-northeast-1	Sì	Sì	Sì	Sì	Sì	Sì	Sì [†]
Canada (Centrale)	ca-central-1	Sì		Sì	Sì	Sì		
Europa (Francoforte)	eu-central-1	Sì	Sì	Sì	Sì	Sì	Sì	Sì [†]
Europa (Irlanda)	eu-west-1	Sì	Sì	Sì	Sì	Sì	Sì	Sì [†]
Europe (London)	eu-west-2	Sì		Sì	Sì	Sì		
Europe (Milan)	eu-south-1			Sì				
Europa (Parigi)	eu-west-3			Sì				
Europa (Stoccolma)	eu-north-1			Sì				
Medio Oriente (Bahrein)	me-south-1			Sì				

Posizione geografica	Codice di ubicazione	Regione d'origine per le flotte gestite (sede unica)	Area geografica per le flotte gestite (con più sedi)	Localizzazione remota per flotte gestite (più sedi)	Flotta ovunque	Coda della sessione di gioco	FlexMatchmaker e set di regole	DDoProtection S (player gateway)
Sud America (San Paolo)	sa-east-1	Sì		Sì	Sì	Sì		
Zona locale di Atlanta	us-east-1-atl-1			Sì				
Zona locale di Chicago	us-east-1-chi-1			Sì				
Zona locale di Dallas*	us-east-1-dfw-1			Sì				
Zona locale 2 di Dallas	us-east-1-dfw-2			Sì				
Zona locale di Denver	us-west-2-den-1			Sì				
Zona locale di Houston	us-east-1-iah-1			Sì				

Posizione geografica	Codice di ubicazione	Regione d'origine per le flotte gestite (sede unica)	Area geografica per le flotte gestite (con più sedi)	Localizzazione remota per flotte gestite (più sedi)	Flotta ovunque	Coda della sessione di gioco	FlexMatch e set di regole	DDoProtezione S (player gateway)
Zona locale di Kansas City	us-east-1-mci-1			Sì				
Zona locale di Los Angeles	us-west-2-lax-1			Sì				
Zona locale di Phoenix	us-west-2-phx-1			Sì				
Zona locale di Lagos, Nigeria	af-south-1-los-1			Sì				

* Disponibile solo per Account AWS chi ha già aderito.

† Regione di origine della flotta valida per la protezione DDoS (player gateway).

Note

Alcune Regioni AWS e tutte le Local Zones non sono abilitate di default per un Account AWS. Devi prima iscriverti a queste Regioni o Local Zones prima di poter distribuire i server di gioco in quelle località. Per ulteriori informazioni sulle regioni che non sono abilitate per impostazione predefinita e su come attivarle, vedi [Gestione Regioni AWS](#) in. Riferimenti generali di AWS Vedi [Guida introduttiva a Local Zones](#) nella AWS Local Zones User Guide. (Le flotte create prima del 28 febbraio 2022 non sono interessate da questo requisito.)

Inoltre, è necessario aggiornare la politica Amazon GameLift Servers dell'amministratore per consentire l'`ec2:DescribeRegions` azione. Per un esempio di politica con le regioni che non sono abilitate per impostazione predefinita, consulta [Esempi di autorizzazioni amministrative](#).

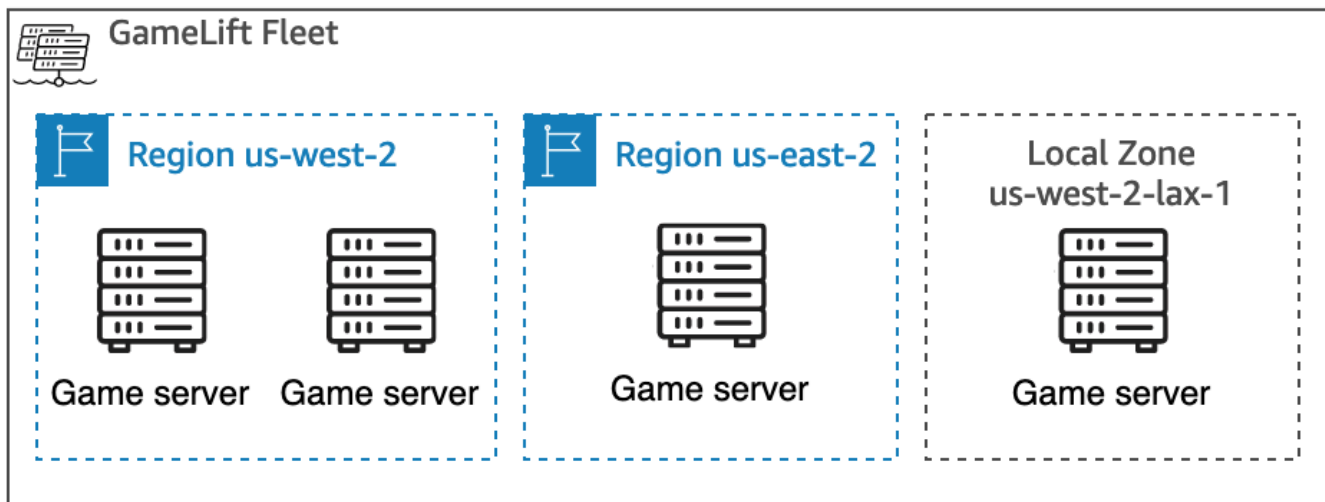
Sedi per l'hosting gestito

Amazon GameLift Servers l'hosting gestito distribuisce flotte di risorse dei server di gioco. Ogni flotta viene creata in una Regione AWS, che è la regione di origine della flotta. La regione di origine di una flotta è indicata nell'Amazon Resource Number (ARN) della flotta.

Puoi implementare una flotta in una sola regione, con risorse di hosting solo nella regione d'origine. In alternativa, puoi implementare una flotta con più sedi, con risorse di hosting in più località geografiche. Una flotta con più sedi ha un'area geografica e una o più sedi remote. Non ci sono limitazioni al modo in cui raggruppi le sedi in una flotta. Quando gestisci la capacità di hosting per una flotta con più sedi, puoi impostare la capacità per ciascuna sede individualmente.

Le località remote per una flotta con più sedi possono essere altre Regioni AWS o Local Zones. Una zona locale è un'estensione di una Regione AWS. Consente di collocare le risorse di elaborazione più vicine agli utenti per fornire un gameplay a bassa latenza. Per ulteriori informazioni, consulta [AWS Zone locali](#). Il codice di posizione di una zona locale è il codice regionale principale seguito da un identificatore di posizione fisico. Ad esempio, il codice per la zona locale di Los Angeles è `us-west-2-lax-1`.

Il diagramma seguente illustra una flotta con più sedi con risorse in due Regioni AWS e una zona locale. La regione di origine della flotta è `us-west-2`, e ha due sedi remote: `us-east-2` Regione e `us-west-2-lax-1` Zona locale.



Oltre alle risorse della flotta, l'hosting gestito con Amazon GameLift Servers può utilizzare anche i seguenti tipi di risorse. Queste risorse vengono create in uno specifico Regione AWS che supporti il tipo di risorsa.

- **Build:** si tratta di un server di gioco costruito per essere ospitato con una flotta EC2 gestita. Crea una risorsa di costruzione nella stessa regione della flotta in cui verrà impiegata.
- **Script:** si tratta di uno script di configurazione con Amazon GameLift Servers Realtime cui ospitare un gioco. Crea una risorsa di script nella stessa regione della flotta in cui verrà distribuita.
- **Definizione del gruppo di container:** si tratta di una configurazione per l'esecuzione di container su una flotta di container gestita. Identifica una o più immagini di container con il software del server di gioco da distribuire alla flotta di container. Crea una definizione di gruppo di container nella stessa regione della flotta in cui verranno distribuiti.
- **Coda delle sessioni di gioco:** questa risorsa elabora le richieste di sessioni di gioco e avvia nuove sessioni di gioco. L'elaborazione avviene nel luogo in Regione AWS cui si trova la coda. Per ridurre la latenza nel processo di posizionamento delle sessioni di gioco, crea una coda geograficamente vicina ai giocatori che la useranno.

Posizioni per qualsiasi luogo Amazon GameLift Servers

Una flotta Amazon GameLift Servers Anywhere è una raccolta di hardware di hosting che fornisci tu. Gestisci tutte le attività sulle tue risorse di hosting, inclusa la distribuzione del software del server di gioco, l'aggiornamento e l'avvio dei processi del server. Crei una flotta Anywhere per connettere il Amazon GameLift Servers servizio con le tue risorse di hosting autogestite. Amazon GameLift

Servers gestisce il posizionamento delle sessioni di gioco: elabora le richieste di iscrizione dei giocatori, individua le risorse di hosting disponibili, avvia nuove sessioni di gioco e fornisce ai client di gioco le informazioni di connessione. Puoi creare una flotta Anywhere in uno qualsiasi dei sistemi che li supportano. Regioni AWS

Puoi aggiungere istanze di hardware di hosting a una flotta Anywhere registrandola. A ogni istanza registrata deve essere associata una posizione personalizzata. Le posizioni personalizzate non sono correlate a Regioni AWS o alle Local Zones. Vengono utilizzate per rappresentare la posizione fisica dell'hardware.

Per ulteriori informazioni sulla creazione di una flotta Anywhere e sul test dell'integrazione con i server di gioco, consulta [Crea un Amazon GameLift Servers Flotta ovunque](#) e [Configura i test locali con Amazon GameLift Servers Anywhere](#).

Sedi per Amazon GameLift Servers FlexMatch

FlexMatch le risorse vengono utilizzate per elaborare le richieste dei giocatori per il matchmaking. Includono una risorsa di configurazione del matchmaking e una risorsa per il set di regole. L'elaborazione avviene nel luogo in Regione AWS cui si trovano le FlexMatch risorse. Per ridurre la latenza nel processo di matchmaking, crea le risorse geograficamente vicino ai giocatori che le useranno. Una configurazione di matchmaking e il set di regole che utilizza devono trovarsi nella stessa configurazione. Regione AWS Puoi creare FlexMatch risorse in qualsiasi sistema Regioni AWS che le supporti.

Per ulteriori informazioni sulla configurazione della tua soluzione di hosting, consulta la [guida FlexMatch per gli Amazon GameLift Servers FlexMatch sviluppatori](#).

Amazon GameLift Servers in Cina

Quando si utilizzano Amazon GameLift Servers risorse nella regione Cina (Pechino), gestita da Sinnet, o nella regione Cina (Ningxia), gestita da NWCD, è necessario disporre di un account separato (AWS Cina). Tieni presente che alcune funzionalità non sono disponibili nelle regioni della Cina. Per ulteriori informazioni sull'utilizzo Amazon GameLift Servers in queste regioni, consulta le seguenti risorse:

- [Amazon Web Services in Cina](#)
- [Amazon GameLift Servers](#) (Guida introduttiva ad Amazon Web Services in Cina)

Endpoint e quote di Amazon GameLift Servers

- [Per gli endpoint di servizio e gli endpoint beacon che puoi utilizzare per connetterti programmaticamente al servizio, consulta Service Endpoints. Amazon GameLift Servers](#)
- [Per le Amazon GameLift Servers quote sull'uso delle risorse o delle operazioni del servizio per account, vedi Service Quotas. AWS Amazon GameLift Servers](#) [Per ulteriori informazioni sulle quote e su come richiedere un aumento, consulta AWS le quote di servizio.](#) Puoi richiedere aumenti delle quote utilizzando la Amazon GameLift Servers console.

Limiti delle API di Amazon GameLift Servers

Le tabelle in questa sezione elencano i limiti di velocità predefiniti per le operazioni API di Amazon GameLift Servers. Questi limiti aiutano a mantenere la stabilità del servizio e a garantire un utilizzo equo. Ogni operazione API ha limiti specifici in base all'impatto sulle risorse e ai modelli di utilizzo.

Per impostazione predefinita, il limite di accelerazione per ogni account cliente è impostato su 500 per le richieste burst e 500 per il traffico normale. Inoltre, ogni API ha una propria coppia di limiti predefinita, come indicato di seguito.

Riferimento ai limiti delle API

La tabella seguente elenca i limiti di velocità predefiniti per le operazioni API di Amazon GameLift Servers. APIs contrassegnati dalla limitazione a livello di risorsa possono essere soggetti a limiti aggiuntivi per evitare problemi con i tasti di scelta rapida del database.

Riferimento ai limiti delle API

Azione API	Limite di aumento delle prestazioni	Limite frequenza	Limitazione a livello di risorsa	Note
AcceptMatch	100	100	No	Il limite è per account.
ClaimGameServer	40	20	No	Il limite è per account.
CreateAlias	4	1	No	Il limite è per account.

Azione API	Limite di aumento delle prestazioni	Limite frequenza	Limitazione a livello di risorsa	Note
CreateBuild	10	1	No	Il limite è per account.
CreateContainerFleet	4	1	No	Il limite è per account.
CreateContainerGroupDefinition	10	1	No	Il limite è per account.
CreateFleet	4	1	No	Il limite è per account.
CreateFleetLocations	4	1	No	Il limite è per account.
CreateGameServerGroup	5	0.1	No	Il limite è per account.
CreateGameSession	50	50	No	Il limite è per account.
CreateGameSessionQueue	4	1	No	Il limite è per account.
CreateLocation	4	1	No	Il limite è per account.
CreateMatchmakingConfiguration	4	1	No	Il limite è per account.
CreateMatchmakingRuleSet	4	1	No	Il limite è per account.
CreatePlayerSession	200	200	No	Il limite è per account.
CreatePlayerSessions	200	200	No	Il limite è per account.
CreateScript	10	1	No	Il limite è per account.

Azione API	Limite di aumento delle prestazioni	Limite frequenza	Limitazione a livello di risorsa	Note
CreateVpcPeeringAuthorization	1	0.1	No	Il limite è per account.
CreateVpcPeeringConnection	1	0.1	No	Il limite è per account.
DeleteAlias	4	1	No	Il limite è per account.
DeleteBuild	10	1	No	Il limite è per account.
DeleteContainerFleet	4	1	No	Il limite è per account.
DeleteContainerGroupDefinition	10	1	No	Il limite è per account.
DeleteFleet	4	1	No	Il limite è per account.
DeleteFleetLocations	4	1	No	Il limite è per account.
DeleteGameServerGroup	5	0.1	No	Il limite è per account.
DeleteGameSessionQueue	4	1	No	Il limite è per account.
DeleteLocation	4	1	No	Il limite è per account.
DeleteMatchmakingConfiguration	4	1	No	Il limite è per account.
DeleteMatchmakingRuleSet	4	1	No	Il limite è per account.
DeleteScalingPolicy	4	1	No	Il limite è per account.

Azione API	Limite di aumento delle prestazioni	Limite frequenza	Limitazione a livello di risorsa	Note
DeleteScript	10	1	No	Il limite è per account.
DeleteVpcPeeringAuthorization	1	0.1	No	Il limite è per account.
DeleteVpcPeeringConnection	1	0.1	No	Il limite è per account.
DeregisterGameServer	40	20	No	Il limite è per account.
DeregisterCompute	10	2	No	Il limite è per account.
DescribeAlias	10	2	No	Il limite è per account.
DescribeBuild	20	5	No	Il limite è per account.
DescribeCompute	10	2	No	Il limite è per account.
DescribeContainerFleet	20	5	No	Il limite è per account.
DescribeContainerGroupDefinition	20	5	No	Il limite è per account.
DescribeContainerGroupPortMappings	20	5	No	Il limite è per account.
Descriva EC2 InstanceLimits	40	15	No	Il limite è per account.
DescribeFleetAttributes	10	2	No	Il limite è per account.

Azione API	Limite di aumento delle prestazioni	Limite frequenza	Limitazione a livello di risorsa	Note
DescribeFleetCapacity	4	1	No	Il limite è per account.
DescribeFleetDeployment	20	5	No	Il limite è per account.
DescribeFleetEvents	10	5	No	Il limite è per account.
DescribeFleetLocationAttributes	10	2	No	Il limite è per account.
DescribeFleetLocationCapacity	10	5	No	Il limite è per account.
DescribeFleetLocationUtilization	10	2	No	Il limite è per account.
DescribeFleetPortSettings	10	2	No	Il limite è per account.
DescribeFleetUtilization	10	2	No	Il limite è per account.
DescribeGameServer	40	20	No	Il limite è per account.
DescribeGameServerGroup	10	2	No	Il limite è per account.
DescribeGameServerInstances	10	2	No	Il limite è per account.

Azione API	Limite di aumento delle prestazioni	Limite frequenza	Limitazione a livello di risorsa	Note
DescribeGameSessionDetails	10	5	No	Il limite è per account. Prendi in considerazione DescribeGameSessions invece l'utilizzo in quanto ha un limite predefinito più elevato e supporta limiti a livello di risorse.
DescribeGameSessionPlacement	75	75	Sì	Il limite è per account. Soggetto a limitazione a livello di risorsa.
DescribeGameSessionQueues	10	2	No	Il limite è per account.
DescribeGameSessions	20	20	Sì	Il limite è per account. Soggetto a limitazione a livello di risorsa.
DescribeInstances	10	2	No	Il limite è per account.
DescribeMatchmaking	200	200	Sì	Il limite è per account. Soggetto a limitazione a livello di risorsa.
DescribeMatchmakingConfigurations	15	7	No	Il limite è per account.
DescribeMatchmakingRuleSets	20	10	No	Il limite è per account.
DescribePlayerSessions	200	200	Sì	Il limite è per account. Soggetto a limitazione a livello di risorsa.
DescribeRuntimeConfiguration	4	1	No	Il limite è per account.

Azione API	Limite di aumento delle prestazioni	Limite frequenza	Limitazione a livello di risorsa	Note
DescribeScalingPolicies	20	20	No	Il limite è per account.
DescribeScript	20	5	No	Il limite è per account.
DescribeVpcPeeringAuthorizations	1	0.1	No	Il limite è per account.
DescribeVpcPeeringConnections	1	0.1	No	Il limite è per account.
GetComputeAccess	4	1	No	Il limite è per account.
GetComputeAuthToken	10	5	No	Il limite è per account.
GetGameSessionLogUrl	4	1	No	Il limite è per account.
GetInstanceAccess	4	1	No	Il limite è per account.
ListAliases	10	2	No	Il limite è per account.
ListBuilds	20	5	No	Il limite è per account.
ListContainerFleets	20	5	No	Il limite è per account.
ListCompute	10	2	No	Il limite è per account.
ListContainerGroupDefinitions	20	5	No	Il limite è per account.
ListContainerGroupDefinitionVersions	20	5	No	Il limite è per account.

Azione API	Limite di aumento delle prestazioni	Limite frequenza	Limitazione a livello di risorsa	Note
ListFleets	10	2	No	Il limite è per account.
ListFleetDeployments	20	5	No	Il limite è per account.
ListGameServerGroups	10	2	No	Il limite è per account.
ListGameServers	10	2	No	Il limite è per account.
ListLocations	10	2	No	Il limite è per account.
ListScripts	20	5	No	Il limite è per account.
ListTagsForResource	20	5	No	Il limite è per account.
PutScalingPolicy	4	1	No	Il limite è per account.
RegisterCompute	10	5	No	Il limite è per account.
RegisterGameServer	40	20	No	Il limite è per account.
RequestUploadCredentials	4	1	No	Il limite è per account.
ResolveAlias	4	1	No	Il limite è per account.
ResumeGameServerGroup	1	0.1	No	Il limite è per account.
SearchGameSessions	10	5	No	Il limite è per account.
StartFleetActions	1	0.1	No	Il limite è per account.
StartGameSessionPlacement	75	75	No	Il limite è per account.

Azione API	Limite di aumento delle prestazioni	Limite frequenza	Limitazione a livello di risorsa	Note
StartMatchBackfill	70	70	No	Il limite è per account.
StartMatchmaking	100	100	No	Il limite è per account.
StopFleetActions	1	0.1	No	Il limite è per account.
StopGameSessionPlacement	20	20	No	Il limite è per account.
StopMatchmaking	100	100	No	Il limite è per account.
SuspendGameServerGroup	1	0.1	No	Il limite è per account.
TagResource	20	5	No	Il limite è per account.
TerminateGameSession	10	2	No	Il limite è per account.
UntagResource	20	5	No	Il limite è per account.
UpdateAlias	4	1	No	Il limite è per account.
UpdateBuild	10	1	No	Il limite è per account.
UpdateContainerGroupDefinition	10	1	No	Il limite è per account.
UpdateContainerFleet	4	1	No	Il limite è per account.
UpdateFleetAttributes	4	1	No	Il limite è per account.
UpdateFleetCapacity	4	1	No	Il limite è per account.

Azione API	Limite di aumento delle prestazioni	Limite frequenza	Limitazione a livello di risorsa	Note
UpdateFleetPortSettings	4	1	No	Il limite è per account.
UpdateGameServer	40	20	No	Il limite è per account.
UpdateGameServerGroup	5	0.1	No	Il limite è per account.
UpdateGameSession	10	2	No	Il limite è per account.
UpdateGameSessionQueue	4	1	No	Il limite è per account.
UpdateMatchmakingConfiguration	4	1	No	Il limite è per account.
UpdateRuntimeConfiguration	4	1	No	Il limite è per account.
UpdateScript	10	1	No	Il limite è per account.
ValidateMatchmakingRuleSet	4	1	No	Il limite è per account.

Limitazione a livello di risorsa

Alcune operazioni API sono soggette a limitazione a livello di risorsa per evitare problemi con i tassi di scelta rapida del database. Pur aumentando i limiti a livello di API per queste operazioni, è possibile che queste vengano comunque limitate a livello di risorsa.

Le seguenti operazioni API sono soggette a limitazione a livello di risorsa:

- DescribeGameSessionPlacement

- DescribeGameSessions
- DescribeMatchmaking
- DescribePlayerSessions

La limitazione a livello di risorsa viene applicata utilizzando tasti di regolazione specifici che includono identificatori di risorse. Ad esempio, DescribeGameSessionPlacement utilizza la chiave «Operation:GameLift/,aws-account-placement-id:» che impone la limitazione per account e la DescribeGameSessionPlacement combinazione di ID di posizionamento. In questo modo si evita che i clienti sovraccarichino il sistema con richieste di risorse specifiche.

Limiti delle API SDK di Amazon GameLift Servers

La tabella seguente elenca i limiti di velocità predefiniti per le operazioni dell'API SDK di Amazon GameLift Servers. Questi limiti ottimizzano le prestazioni e prevengono il conflitto di risorse negli ambienti dei server di gioco. La comprensione di questi limiti è importante per un'efficace integrazione lato server con Amazon GameLift Servers.

I limiti dell'SDK del server vengono applicati utilizzando chiavi di limitazione specifiche che includono identificatori di account e processi. Alcuni limiti sono per giocatore per prevenire abusi da parte dei singoli giocatori.

Note

Questa tabella è soggetta a modifiche. La fonte autorevole per i limiti attuali è la configurazione SDC interna. Contatta l' AWS assistenza per la maggior parte delle up-to-date informazioni, se necessario.

Riferimento ai limiti dell'API Server SDK

Azione API	Limite di aumento delle prestazioni	Limite frequenza	Chiave di limitazione	Note
AcceptPlayerSession	3	1	aws-account-and-process-id-and-player-id	Il limite è per giocatore e per processo. Non dovrebbe mai essere necessario un aumento del limite in quanto è per giocatore.
RemovePlayerSession	3	1	aws-account-and-process-id-and-player-id	Il limite è per giocatore e per processo. Non dovrebbe mai essere necessario un aumento del limite in quanto è per giocatore.
ActivateGameSession	3	1	aws-account-and-process-id	Il limite è per processo del server. Non dovrebbe mai essere necessario un aumento del limite in quanto dipende dal processo.
ActivateHostProcessV2	3	1	aws-account-and-process-id	Il limite è per processo del server. Non dovrebbe mai essere necessario un aumento del limite in quanto dipende dal processo.
StartMatchBackfill	3	1	aws-account-and-process-id	Il limite è per processo del server. Non dovrebbe mai essere necessario un aumento del limite in quanto dipende dal processo.

Azione API	Limite di aumento delle prestazioni	Limite frequenza	Chiave di limitazione	Note
DescribePayerSessions	10	5	aws-account-and-process-id	Il limite è per processo del server. Non dovrebbe mai essere necessario un aumento del limite in quanto dipende dal processo.
GetComputeCertificate	3	0.1	aws-account-and-process-id	Il limite è per processo del server. Non dovrebbe mai essere necessario un aumento del limite in quanto dipende dal processo.
GetCustomRoleCredentials	3	0.1	aws-account-and-process-id	Il limite è per processo del server. Non dovrebbe mai essere necessario un aumento del limite in quanto dipende dal processo.
InitSDK	3	0.1	aws-account-and-process-id	Il limite è per processo del server. Non dovrebbe mai essere necessario un aumento del limite in quanto dipende dal processo.
RefreshHostProcess	3	0.1	aws-account-and-process-id	Il limite è per processo del server. Non dovrebbe mai essere necessario un aumento del limite in quanto dipende dal processo.

Azione API	Limite di aumento delle prestazioni	Limite frequenza	Chiave di limitazione	Note
StopMatchmaking	3	1	aws-account-and-process-id	Il limite è per processo del server. Non dovrebbe mai essere necessario un aumento del limite in quanto dipende dal processo.
ProcessEnding	3	1	aws-account-and-process-id	Il limite è per processo del server. Non dovrebbe mai essere necessario un aumento del limite in quanto dipende dal processo.
ProcessReady	3	1	aws-account-and-process-id	Il limite è per processo del server. Non dovrebbe mai essere necessario un aumento del limite in quanto dipende dal processo.
UpdateGameSession	3	1	aws-account-and-process-id	Il limite è per processo del server. Non dovrebbe mai essere necessario un aumento del limite in quanto dipende dal processo.
UpdateHostProcess	3	1	aws-account-and-process-id	Il limite è per processo del server. Non dovrebbe mai essere necessario un aumento del limite in quanto dipende dal processo.

Azione API	Limite di aumento delle prestazioni	Limite frequenza	Chiave di limitazione	Note
GetGameSessionId	10	5	aws-account-and-process-id	Il limite è per processo del server. Non dovrebbe mai essere necessario un aumento del limite in quanto dipende dal processo.
GetTerminationTime	3	1	aws-account-and-process-id	Il limite è per processo del server. Non dovrebbe mai essere necessario un aumento del limite in quanto dipende dal processo.
UpdatePlayerSessionCreationPolicy	3	1	aws-account-and-process-id	Il limite è per processo del server. Non dovrebbe mai essere necessario un aumento del limite in quanto dipende dal processo.

Note

Questi sono i limiti predefiniti. Se riscontri una limitazione su questi APIs, potrebbe essere dovuta alla limitazione a livello di account. Per richiedere un aumento del limite, contatta l'AWS assistenza.

I limiti per processo e per giocatore sono progettati per adattarsi ai normali modelli di utilizzo e raramente dovrebbero essere modificati. Se riscontri problemi con questi limiti APIs, esamina la tua implementazione per potenziali opportunità di ottimizzazione prima di richiedere aumenti dei limiti.

La limitazione a livello di account può essere applicata anche attraverso queste regole aggiuntive:

- SdkWebSocketAccountLevelHighUsage/, aws-account
- SdkWebSocket/AccountLevelThrottle, aws-account

Amazon GameLift Servers Fari ping UDP

I beacon ping UDP forniscono un modo per misurare la latenza di rete tra i dispositivi dei giocatori e le posizioni di hosting. Amazon GameLift Servers Con i ping beacon, puoi raccogliere dati di latenza accurati per prendere decisioni informate sul posizionamento dei server di gioco e migliorare il matchmaking dei giocatori in base ai requisiti di latenza.

Come funzionano i ping beacon UDP

Amazon GameLift Servers fornisce endpoint UDP fissi (ping beacon) in ogni posizione di hosting in cui è possibile distribuire server di gioco. Poiché la maggior parte dei server di gioco comunica tramite UDP, la misurazione della latenza con i beacon ping UDP fornisce risultati più accurati rispetto all'utilizzo dei ping ICMP. I dispositivi di rete spesso gestiscono i pacchetti ICMP in modo diverso dai pacchetti UDP, il che può portare a misurazioni della latenza che non riflettono le prestazioni reali dei giocatori.

Con i ping beacon UDP, il tuo client di gioco può inviare messaggi UDP a questi endpoint e ricevere risposte asincrone, ottenendo misurazioni della latenza che rappresentano meglio le condizioni effettive del traffico di gioco tra il dispositivo di un giocatore e le potenziali località di hosting. Gli endpoint sono permanenti e rimangono disponibili fintanto che supporta l'hosting di giochi in quella posizione. Amazon GameLift Servers

Casi d'uso comuni dei beacon ping UDP

Puoi utilizzare i beacon ping UDP in diversi modi per ottimizzare l'esperienza di rete del gioco.

Scelta delle location di hosting ottimali

Raccogli i dati sulla latenza in diverse aree geografiche per identificare le migliori posizioni primarie e di backup per ospitare i server di gioco per la tua base di giocatori.

Posizionamento delle sessioni di gioco in base alla latenza dei giocatori

Includi i dati sulla latenza dei giocatori quando richiedi nuove sessioni di gioco per aiutarti a scegliere le località che offrono l'esperienza di latenza più bassa.

Ottimizzazione del matchmaking in base alla latenza

Fornisci dati sulla latenza dei giocatori quando richiedi il matchmaking per aiutare ad abbinare giocatori con profili di latenza simili e a collocare le sessioni di gioco in posizioni ottimali per i giocatori abbinati.

Note

Quando crei richieste di matchmaking, non dovresti fornire informazioni sulla latenza a località in cui non disponi di flotte. In tal caso, Amazon GameLift Servers potresti provare a collocare la sessione di gioco in luoghi in cui la flotta non è disponibile, con conseguenti errori nelle richieste di matchmaking.

Ottenere endpoint beacon

Per recuperare le informazioni sul dominio e sulla porta del ping beacon per le Amazon GameLift Servers posizioni, utilizza l'operazione API. [ListLocations](#) Il set di posizioni restituito da questa API dipende dalla regione specificata al Regione AWS momento della chiamata (o dalla regione predefinita se non ne specifichi una).

Quando chiami da:

- Una regione di origine di una flotta che supporta più sedi: l'API restituisce informazioni per tutte le sedi di hosting
- Una regione di origine di un parco veicoli che supporta un'unica ubicazione: l'API restituisce le informazioni relative a quella località

Tieni presente che se chiami questa API utilizzando una posizione che può essere solo una posizione remota in un parco di sedi multiple, l'API restituirà un errore perché quel tipo di posizione non dispone di un endpoint di servizio.

Consulta la tabella delle località supportate in [AWS Località supportate](#) per identificare le aree geografiche che supportano flotte con una o più sedi.

Esempio

```
aws gamelift list-locations --region ap-northeast-2
```

Questo Regione AWS supporta flotte con più sedi, pertanto verranno restituite più sedi. Ecco un esempio di uno dei valori restituiti:

```
[...]
{
  "LocationName": "ap-northeast-1",
  "PingBeacon": {
    "UDPEndpoint": {
      "Domain": "gamelift-ping.ap-northeast-1.api.aws",
      "Port": 7770
    }
  }
}
```

Important

Memorizza nella cache le informazioni del beacon ping anziché chiamare `ListLocations` prima di ogni misurazione della latenza. Le informazioni sul dominio e sulla porta sono statiche e l'API non è progettata per richieste ad alto volume.

Implementazione della misurazione della latenza

Segui queste best practice quando implementi le misurazioni della latenza utilizzando i beacon ping UDP:

1. Memorizza le informazioni sui ping beacon utilizzando uno di questi approcci:

- Codifica gli endpoint nel tuo client di gioco.
- Memorizza le informazioni nel backend del gioco.
- Implementa un meccanismo di aggiornamento periodico (giornaliero/settimanale) per aggiornare le informazioni.

2. Invia messaggi ping UDP:

- Inserisci quello che vuoi nel corpo del messaggio, purché non sia vuoto, e mantieni i messaggi al di sotto della dimensione massima di 300 byte.
- Rispetta i seguenti limiti di velocità per ogni località:
 - 3 transazioni al secondo (TPS) per combinazione univoca di indirizzo IP e porta del mittente
 - 1000 TPS per indirizzo IP univoco del mittente

3. Calcola la latenza:

- Invia più ping a ciascuna posizione per calcolare una latenza media.
- Valuta la possibilità di inviare ping simultanei a più sedi per risultati più rapidi.
- Se necessario, utilizzate la logica di ripetizione per inviare nuovi pacchetti per qualsiasi pacchetto che non sia stato restituito entro breve tempo (in genere 1-3 secondi), poiché UDP non offre una consegna garantita al 100%.
- Calcola la differenza di tempo tra l'invio di un messaggio e la ricezione della risposta.
- Se una gran parte dei ping UDP verso una posizione non ottiene costantemente una risposta, calcola la latenza utilizzando i ping ICMP ai nostri endpoint di [Amazon GameLift Servers servizio](#) standard come fallback.

Tip

Ti consigliamo di includere la porta 7770 ovunque documenti l'elenco delle porte che i giocatori devono avere aperte sulla rete locale. Questo è un altro motivo per cui dovresti disporre di un sistema di riserva per misurare la latenza (usando ICMP, ad esempio) nel caso in cui questa porta sia bloccata.

Esempi di codice

Ecco alcuni semplici esempi che mostrano come inviare ping UDP e calcolare la latenza.

C++

```
#include <iostream>
#include <cstring>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <unistd.h>
#include <chrono>

int main() {
    // Replace with Amazon GameLift Servers UDP ping beacon domain for your desired
    location
    const char* domain = "gamelift-ping.ap-south-1.api.aws";
```

```
const int port = 7770;
const char* message = "Ping";           // Your message
const int num_pings = 3;                 // Number of pings to send

// Create socket
int sock = socket(AF_INET, SOCK_DGRAM, 0);
if (sock < 0) {
    std::cerr << "Error creating socket" << std::endl;
    return 1;
}

// Resolve domain name to IP address
struct hostent* host = gethostbyname(domain);
if (host == nullptr) {
    std::cerr << "Error resolving hostname" << std::endl;
    close(sock);
    return 1;
}

// Set up the server address structure
struct sockaddr_in server_addr;
std::memset(&server_addr, 0, sizeof(server_addr));
server_addr.sin_family = AF_INET;
server_addr.sin_port = htons(port);
std::memcpy(&server_addr.sin_addr, host->h_addr, host->h_length);

// Set socket timeout
struct timeval tv;
tv.tv_sec = 1; // 1 second timeout
tv.tv_usec = 0;
if (setsockopt(sock, SOL_SOCKET, SO_RCVTIMEO, &tv, sizeof(tv)) < 0) {
    std::cerr << "Error setting socket timeout" << std::endl;
    close(sock);
    return 1;
}

double total_latency = 0;
int successful_pings = 0;

for (int i = 0; i < num_pings; ++i) {
    auto start = std::chrono::high_resolution_clock::now();

    // Send the message
    ssize_t bytes_sent = sendto(sock, message, std::strlen(message), 0,
```



```
        (struct sockaddr*)&server_addr,
sizeof(server_addr));

    if (bytes_sent < 0) {
        std::cerr << "Error sending message" << std::endl;
        continue;
    }

    // Receive response
    char buffer[1024];
    socklen_t server_addr_len = sizeof(server_addr);
    ssize_t bytes_received = recvfrom(sock, buffer, sizeof(buffer), 0,
        (struct sockaddr*)&server_addr,
&server_addr_len);

    auto end = std::chrono::high_resolution_clock::now();

    if (bytes_received < 0) {
        std::cerr << "Error receiving response or timeout" << std::endl;
    } else {
        std::chrono::duration<double, std::milli> latency = end - start;
        total_latency += latency.count();
        successful_pings++;

        std::cout << "Received response, latency: " << latency.count() << " ms"
<< std::endl;
    }

    // Wait a bit before next ping
    usleep(1000000); // 1s
}

// Close the socket
close(sock);

if (successful_pings > 0) {
    double avg_latency = total_latency / successful_pings;
    std::cout << "Average latency: " << avg_latency << " ms" << std::endl;
} else {
    std::cout << "No successful pings" << std::endl;
}

return 0;
```

```
}
```

C#

```
using System;
using System.Net;
using System.Net.Sockets;
using System.Text;
using System.Diagnostics;
using System.Threading.Tasks;

class UdpLatencyTest
{
    static async Task Main()
    {
        // Replace with Amazon GameLift Servers UDP ping beacon domain for your
        // desired location
        string domain = "gamelift-ping.ap-south-1.api.aws";
        int port = 7770;
        string message = "Ping";           // Your message
        int numPings = 3;                  // Number of pings to send
        int timeoutMs = 1000;              // Timeout in milliseconds

        await MeasureLatency(domain, port, message, numPings, timeoutMs);
    }

    static async Task MeasureLatency(string domain, int port, string message, int
numPings, int timeoutMs)
    {
        using (var udpClient = new UdpClient())
        {
            try
            {
                // Resolve domain name to IP address
                IPAddress[] addresses = await Dns.GetHostAddressesAsync(domain);
                if (addresses.Length == 0)
                {
                    Console.WriteLine("Could not resolve domain name.");
                    return;
                }

                IPEndPoint endPoint = new IPEndPoint(addresses[0], port);
                byte[] messageBytes = Encoding.UTF8.GetBytes(message);
```

```
// Set receive timeout
udpClient.Client.ReceiveTimeout = timeoutMs;

double totalLatency = 0;
int successfulPings = 0;
var stopwatch = new Stopwatch();

for (int i = 0; i < numPings; i++)
{
    try
    {
        stopwatch.Restart();

        // Send message
        await udpClient.SendAsync(messageBytes, messageBytes.Length,
endPoint);

        // Wait for response
        UdpReceiveResult result = await
ReceiveWithTimeoutAsync(udpClient, timeoutMs);

        stopwatch.Stop();
        double latency = stopwatch.Elapsed.TotalMilliseconds;

        totalLatency += latency;
        successfulPings++;

        string response = Encoding.UTF8.GetString(result.Buffer);
        Console.WriteLine($"Ping {i + 1}: {latency:F2}ms - Response:
{response}");
    }
    catch (SocketException ex)
    {
        Console.WriteLine($"Ping {i + 1}: Failed - {ex.Message}");
    }
    catch (TimeoutException)
    {
        Console.WriteLine($"Ping {i + 1}: Timeout");
    }

    // Wait before next ping
    await Task.Delay(1000); // 1s between pings
}
```

```

        if (successfulPings > 0)
        {
            double averageLatency = totalLatency / successfulPings;
            Console.WriteLine($"Summary:");
            Console.WriteLine($"Successful pings: {successfulPings}/{
numPings}");
            Console.WriteLine($"Average latency: {averageLatency:F2}ms");
        }
        else
        {
            Console.WriteLine($"No successful pings");
        }
    }
    catch (Exception ex)
    {
        Console.WriteLine($"Error: {ex.Message}");
    }
}

static async Task<UdpReceiveResult> ReceiveWithTimeoutAsync(UdpClient client,
int timeoutMs)
{
    using var cts = new System.Threading.CancellationTokenSource(timeoutMs);
    try
    {
        return await client.ReceiveAsync().WaitAsync(cts.Token);
    }
    catch (OperationCanceledException)
    {
        throw new TimeoutException("Receive operation timed out");
    }
}
}

```

Python

```

import socket
import time
import statistics
from datetime import datetime

```

```
def udp_ping(host, port, timeout=2):
    """
    Send a UDP ping and return the round trip time in milliseconds.
    Returns None if timeout occurs.
    """
    sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
    sock.settimeout(timeout)
    message = b'ping'

    try:
        # Resolve hostname first
        try:
            socket.gethostbyname(host)
        except socket.gaierror as e:
            print(f"Could not resolve hostname: {e}")
            return None

        start_time = time.time()
        sock.sendto(message, (host, port))

        # Wait for response
        data, server = sock.recvfrom(1024)
        end_time = time.time()

        # Calculate round trip time in milliseconds
        rtt = (end_time - start_time) * 1000
        return rtt

    except socket.timeout:
        print(f"Request timed out")
        return None
    except Exception as e:
        print(f"Error: {type(e).__name__}: {e}")
        return None
    finally:
        sock.close()

def main():
    # Replace with Amazon GameLift Servers UDP ping beacon domain for your desired
    location
    host = "gamelift-ping.ap-south-1.api.aws"
    port = 7770
    num_pings = 3
    latencies = []
```

```

print(f"\nPinging {host}:{port} {num_pings} times...")
print(f"Start time: {datetime.now().strftime('%Y-%m-%d %H:%M:%S')}\n")

for i in range(num_pings):
    print(f"Ping {i+1}:")
    rtt = udp_ping(host, port)

    if rtt is not None:
        print(f"Response from {host}: time={rtt:.2f}ms")
        latencies.append(rtt)

    # Wait 1 second between pings
    if i < num_pings - 1:
        time.sleep(1)
    print()

# Calculate and display statistics
print("-" * 50)
print(f"Ping statistics for {host}:")
print(f"    Packets: Sent = {num_pings}, Received = {len(latencies)}, "
      f"Lost = {num_pings - len(latencies)} "
      f"f"({((num_pings - len(latencies)) / num_pings * 100):.1f}% loss)")

if latencies:
    print("\nRound-trip latency statistics:")
    print(f"    Minimum = {min(latencies):.2f}ms")
    print(f"    Maximum = {max(latencies):.2f}ms")
    print(f"    Average = {statistics.mean(latencies):.2f}ms")

print(f"\nEnd time: {datetime.now().strftime('%Y-%m-%d %H:%M:%S')}")

if __name__ == "__main__":
    main()

```

FAQs sui protocolli supportati

[Amazon GameLift Servers aggiunge il supporto SDK per il protocollo Smithy RPC v2 CBOR oltre al protocollo JSON esistente.](#) AWS Il protocollo Smithy RPC v2 CBOR è più performante di JSON per la maggior parte dei carichi di lavoro. AWS AWS SDKs darà la priorità al protocollo più performante per ogni lingua.

Cosa sono i protocolli AWS CBOR JSON e Smithy RPC v2 e in che cosa differiscono?

Un protocollo cablato funge da mediatore tra un cliente e il servizio. L' AWS SDK gestisce la serializzazione (conversione di un oggetto nel formato wire) e la deserializzazione (riconversione del formato wire in un oggetto) tra il client e il server in modo trasparente per entrambi i protocolli.

AWS JSON - JSON è uno dei formati di dati più utilizzati e accettati per la comunicazione tra sistemi. Questo servizio utilizza AWS JSON come mezzo di comunicazione tra un client AWS SDK e il server. Una richiesta HTTP per un'operazione API di servizio accetta l'input sotto forma di JSON, l'operazione viene eseguita e la risposta viene restituita al client SDK sotto forma di JSON.

Smithy RPC v2 CBOR - Smithy RPC v2 CBOR (Concise Binary Object Representation) è un protocollo sviluppato da Amazon per una trasmissione dei dati più efficiente. È un formato di dati progettato per mantenere compatte le dimensioni dei messaggi e per consentire al formato di evolversi senza richiedere la negoziazione della versione tra i sistemi. La differenza fondamentale rispetto a AWS JSON è che AWS JSON utilizza testo leggibile dall'uomo (ad esempio `{"name": "John"}`), mentre Smithy RPC v2 CBOR utilizza un formato binario (sequenze di 1 e 0). Questo approccio binario crea pacchetti di dati più piccoli che viaggiano più velocemente sulla rete e riduce i tempi di serializzazione poiché i computer elaborano direttamente i dati binari.

Come posso iniziare a usare il protocollo CBOR Smithy RPC v2?

Per iniziare, utilizza l'ultima versione AWS SDK/CLI per utilizzare il protocollo CBOR Smithy RPC v2 per questo servizio. L'SDK utilizzerà automaticamente il protocollo Smithy RPC v2 CBOR quando disponibile. Aggiorna il tuo AWS SDK alla versione specificata o a qualsiasi versione successiva.

A partire dal 16/04/2020, i seguenti AWS SDKs utilizzeranno il protocollo Smithy RPC v2 CBOR, mentre gli altri continueranno a utilizzare il protocollo JSON: SDKs AWS

- AWS SDK per C++
- AWS SDK per Go V2
- AWS SDK per Java 2.x
- AWS SDK per .NET v4
- AWS Strumenti per v5 PowerShell
- AWS SDK per Swift
- AWS SDK per Kotlin

Quali sono i rischi dell'abilitazione del protocollo Smithy RPC v2 CBOR?

Il supporto aggiuntivo di Smithy RPC v2 CBOR nel dovrebbe essere completamente trasparente e semplice per i SDKs clienti. Tuttavia, se si utilizza un'implementazione personalizzata dell'SDK o una combinazione di client personalizzati e AWS SDK, ciò dipende dal comportamento di serializzazione AWS specifico di JSON, AWS potrebbe essere incompatibile con il protocollo CBOR Smithy RPC v2.

Cosa succede se la mia soluzione non supporta Smithy RPC v2 CBOR?

Per utilizzare il protocollo AWS JSON, esegui il downgrade del tuo SDK a una versione rilasciata prima del 16/04/2026.

Quali AWS regioni supportano il protocollo Smithy RPC v2 CBOR?

[Il protocollo CBOR Smithy RPC v2 è supportato in tutte le regioni in cui questo servizio è disponibile.AWS](#)

Quali miglioramenti della latenza posso aspettarmi eseguendo l'aggiornamento a versioni SDK che supportano Smithy RPC v2 CBOR?

Smithy RPC v2 CBOR offre una latenza inferiore fino al 13% e una riduzione fino al 15% delle dimensioni del payload rispetto a JSON. AWS La codifica binaria elimina il sovraccarico dovuto

all'analisi del testo in JSON e all'escape delle stringhe, riducendo sia i tempi di trasferimento in rete che i costi di serializzazione.

Il protocollo JSON sarà obsoleto? AWS

AWS Il protocollo JSON continuerà a essere supportato dalle versioni precedenti di SDKs. Inoltre, alcuni linguaggi specifici SDKs continueranno a utilizzare AWS JSON in quanto è più performante di Smithy RPC v2 CBOR grazie all'implementazione JSON disponibile per il linguaggio che è più veloce della sua implementazione CBOR.

Dove posso trovare ulteriori informazioni su Smithy RPC v2 CBOR?

[Puoi trovare ulteriori informazioni sul protocollo JSON nei protocolli AWS JSON 1.0 e AWS JSON 1.1 nella documentazione di Smithy e sul protocollo CBOR di Smithy RPC v2 presso AWS Smithy RPC v2 CBOR nella documentazione di Smithy.](#)

Note di rilascio di Amazon GameLift Servers

Le note di rilascio di Amazon GameLift Servers forniscono informazioni sulle nuove funzionalità, aggiornamenti e correzioni relative al servizio.

Versioni SDK

Le tabelle seguenti elencano tutte le Amazon GameLift Servers versioni con informazioni sulla versione dell'SDK. Non è necessario utilizzare strumenti simili SDKs per le integrazioni tra server di gioco e client. Tuttavia, le versioni precedenti di un SDK potrebbero non supportare completamente le funzionalità più recenti di un altro SDK.

Amazon GameLift Servers SDKs e i plugin sono open source. Per informazioni, consulta [Ottieni strumenti Amazon GameLift Servers di sviluppo](#). Per ottenere le versioni più recenti, consulta l'[Amazon GameLift Servers GitHub organizzazione](#).

Versione attuale

1. Versione
 2. Successiv

1. **Time**
 2. **Cost**
 3. **Quality**
 4. **Service**
 5. **Client**
 6. **Team**
 7. **City**
 8. **++**
 9. **Real**

1952
 versione
 successiva

versione
successiva

Versioni SDK

	Visual Studio
	C++
	C#
	Python
	Java
	JavaScript
	PHP
	Ruby
	Go
	Kotlin
	Swift
	Objective-C
	C++
	C#
	Python
	Java
	JavaScript
	PHP
	Ruby
	Go
	Kotlin
	Swift
	Objective-C
	C++
	C#
	Python
	Java

Werkzeuge

Technische Details

cliente

creale

1. $\frac{1}{2} \log 2$



Jsreal

351

versione

successiv

00008151

411

versione

successiv



34

versione

Successiv

133

520000251

510

Versione

successiv

13

	RealtimeS
	K
	Utilità
	cliente
	Creale
	City
	H+
	J)real
	2-1-1951
	versione
	successiv
	2-1-1951
	successiv
	2-1-1951
	successiv
	2-1-1951
	Versione
	Successiv
	2-1-1931
	Successiv

	RealtimeS
	K
	Utilità
	cliente
	Creale
	City
	H+
	In)real
	61001930
	Successiv
	61001942
	versione
	Successiv
	61001841
	versione
	Successiv
	61001712
	versione
	Successiv
	60200612
	versione
	Successiv

Versioni SDK

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

50200-1

	Miglior tempo
	Foto
	Servizi
	cliente
	Crea le
	HQ
	++
	In real
31-01-2020	-3
	Successiv
31-01-2020	-0
	Successiv
31-01-2020	-2
	versione
	Successiv
31-01-2020	-2
	versione
	Successiv
31-01-2020	-0
	versione
	Successiv

	RealtimeS
	K
	Utilità
	cliente
	Creale
	City
	H+
	J)real
	B1168-2
	versione
	successiv
	B1168-1
	versione
	successiv
	B1169-0
	successiv
	B1170-2
	versione
	successiv
	B1170-2
	versione
	successiv

Verfahrenstechnik

FOOD

Selbsttätigkeit

cliente

Creale

● 井戸

→ **+**

Int)real

303210-1

10

Versione

successiv

自

343120-1

Successiv

343008-2

Versione

Successiv

三三

3430.04-1

versione

Successiv

31

3400-0

51

Versione

successiv

133

Wissenschaften



Univ. of Illinois

cliente

Creale

大和

→ **++**

Int)real

3.410019-1

versione

Successiv

13

3.4.0101-1



versione

Successiv

13

3.4101.001

versione

Successiv

131

3.40.001

versione

Successiv

131

Wissenschaften



Introduction

cliente

Creale

● 非(Not)

Jsreal

03301001

11

versione

successiv

131

373904-2

11

versione

successiv

131

3.3.0



versione

successiv

131

173.002-0

versione

Successiv

131

Wissenschaften

FDCK

Özellikler

cliente

 **reale**

● 大和

→ **++**

Jsreal

3.3.00 2-1

versione

Successiv

13

3628109-2

• **10**

Versione

Successiv

131

34281706-1

versione

successiv

131

34281705-1

Versione

successiv

134

1.1.10	TimeS
1.1.10	cliente
1.1.10	creale
1.1.10	toy
1.1.10	real
1.1.10	2-1
1.1.10	versione
1.1.10	successiv
1.1.10	
1.1.10	2-0
1.1.10	versione
1.1.10	successiv
1.1.10	
1.1.10	9-0
1.1.10	versione
1.1.10	successiv
1.1.10	
1.1.10	8-1
1.1.10	versione
1.1.10	successiv
1.1.10	
1.1.10	1-1
1.1.10	versione
1.1.10	successiv
1.1.10	

Wissenschaften

Univis

cliente

Creale

● 井戸

Jsreal

3017504-1

• **10**

Versione

successiv

133

3.11.502-2

versione

successiv

13

301601 1-1

versione

Successiv

301607 0-1

versione

successiv

131

Versioni
3.16.0
3.15.0
3.14.0
3.13.0
3.12.0
3.11.0
3.10.0
3.9.0
3.8.0
3.7.0
3.6.0
3.5.0
3.4.0
3.3.0
3.2.0
3.1.0
3.0.0
2.9.0
2.8.0
2.7.0
2.6.0
2.5.0
2.4.0
2.3.0
2.2.0
2.1.0
2.0.0
1.9.0
1.8.0
1.7.0
1.6.0
1.5.0
1.4.0
1.3.0
1.2.0
1.1.0
1.0.0
0.9.0
0.8.0
0.7.0
0.6.0
0.5.0
0.4.0
0.3.0
0.2.0
0.1.0

Note di rilascio

Le seguenti note di rilascio sono in ordine cronologico, con gli aggiornamenti più recenti elencati per primi. Amazon GameLift Servers è stato rilasciato per la prima volta nel 2016. Per le note di rilascio precedenti a quelle elencate qui, consulta i collegamenti per le date di rilascio in [Versioni SDK](#).

29 aprile 2026: Amazon GameLift Servers aggiunge un nuovo supporto per la visualizzazione delle mappature dei porti dei container

Amazon GameLift Servers ha lanciato una nuova API pubblica per le flotte di container, `DescribeContainerGroupPortMappings`. Questa operazione recupera le mappature delle porte per tutti i container in un gruppo di container, elencando come ogni porta container è mappata a una porta di connessione sull'istanza della flotta. Supporta il recupero delle mappature delle porte sia per i server di gioco che per i gruppi di contenitori per istanza.

Questa nuova funzionalità risponde alla necessità di scoprire le mappature delle porte per i contenitori oltre la porta del server di gioco. In precedenza, ai clienti era richiesto di accedere in remoto al proprio computer per recuperare tutte le altre mappature delle porte. La nuova API è disponibile tramite la Amazon GameLift Servers console e l'SDK. AWS CLI AWS

Ulteriori informazioni:

- [Visualizza le mappature delle porte del contenitore](#), Guida per Amazon GameLift Servers agli sviluppatori
- [DescribeContainerGroupPortMappings](#), Riferimento Amazon GameLift Servers API

2 aprile 2026: Amazon GameLift Servers aggiunge il nome di calcolo ai dati delle sessioni di gioco, ai nuovi campi degli eventi e ai miglioramenti della console

Questa versione semplifica l'installazione e l'integrazione dei giochi da tavolo Amazon GameLift Servers, in base al feedback dei clienti. Questi aggiornamenti aiutano i clienti a iniziare e gestire più facilmente le sessioni di gioco all'interno delle flotte.

Amazon GameLift Servers ora restituisce un `ComputeName` campo nelle risposte API delle sessioni di gioco. Puoi utilizzare questo campo per identificare quale computer ospita una sessione di gioco, senza incrociare gli indirizzi IP. Questo campo è disponibile per flotte EC2 gestite, flotte di container gestite e flotte Anywhere. I `GameSessionLocation` campi `ComputeName` and sono ora inclusi anche negli eventi. [PlacementFulfilled](#) [MatchmakingSucceeded](#)

Amazon GameLift Servers Link di riferimento API:

- [GameSession](#)

Questa versione include anche diversi miglioramenti Amazon GameLift Servers alla console. Con questi aggiornamenti, ora puoi:

- Carica le build direttamente: oltre al metodo di importazione di Amazon S3, puoi selezionare un file di build e Amazon GameLift Servers caricarlo direttamente dalla console.
- Clona configurazioni del parco veicoli: clona una configurazione gestita esistente del parco veicoli EC2 per creare una nuova flotta.
- Connettiti a istanze e contenitori: utilizza SSM per connetterti direttamente dalla sessione di gioco, dalla sessione del giocatore e dalle pagine dei dettagli di calcolo, inclusi i nuovi comandi di accesso per i contenitori.

- Visualizza i dati di calcolo e di istanza: ora vengono visualizzate le tabelle delle sessioni di gioco ComputeName e InstanceId le colonne per le flotte gestite di EC2 e container.
- Visualizza i dettagli di calcolo della flotta di container: le flotte di container ora includono tabelle di calcolo e visualizzazioni dei dettagli di calcolo.
- Visualizza le posizioni dei registri delle sessioni di gioco: la pagina dei dettagli della sessione di gioco ora mostra Amazon S3 e il gruppo di CloudWatch log URIs Amazon ARNs per flotte di container.
- Aggiorna dati: i pulsanti di aggiornamento sono ora disponibili in tutte le pagine dell'elenco e nelle visualizzazioni di dettaglio.

Ulteriori informazioni:

- [Console Amazon GameLift Servers](#)
- [PlacementFulfilled](#), Amazon GameLift Servers Guida per gli sviluppatori
- [MatchmakingSucceeded](#), Guida Amazon GameLift Servers FlexMatch per gli sviluppatori
- [Connessione a un'istanza di flotta di container](#), Guida per Amazon GameLift Servers gli sviluppatori

27 marzo 2026: Amazon GameLift Servers ora supporta anche le istanze EC2 di nuova generazione Regioni AWS

Amazon GameLift Servers ora supporta in aggiunta la selezione esistente di istanze Amazon EC2 Regioni AWS, offrendoti una maggiore flessibilità per ospitare server di gioco più vicini ai tuoi giocatori. Questa espansione copre le istanze delle seguenti famiglie:

- [Uso generico](#) (serie M)
- [Ottimizzato per il calcolo \(serie C\)](#)
- [Memoria ottimizzata](#) (serie R)

Questi tipi di istanze sono ora disponibili per l'uso con flotte EC2 Amazon GameLift Servers gestite e flotte di container gestite. Non sono necessarie modifiche alla configurazione della flotta: è sufficiente distribuire le flotte nelle nuove regioni supportate utilizzando gli stessi tipi di istanze e le stesse impostazioni che usi oggi. Sono disponibili in tutte Regioni AWS le regioni supportate da Amazon GameLift Servers, ad eccezione delle regioni della AWS Cina.

Ulteriori informazioni:

- [Amazon GameLift ServersPrezzi delle istanze](#)
- [Amazon GameLift Serverssedi di assistenza](#), Guida Amazon GameLift Servers per gli sviluppatori
- [CreateFleet](#), Riferimento Amazon GameLift Servers API
- [CreateContainerFleet](#), Riferimento Amazon GameLift Servers API

4 marzo 2026: Amazon GameLift Servers ora offre la protezione DDoS per EC2 e flotte di container basate su Linux su SDKv5

Amazon GameLift Serversora offre la protezione distributed-denial-of-service (DDoS) per EC2 e flotte di container basati su Linux. SDKv5 La rete Player Gateway Proxy Relay Network si trova nella stessa posizione delle istanze del server di gioco e consente la convalida del traffico, la limitazione della velocità per giocatore e l'offuscamento dell'indirizzo IP del server di gioco, il tutto con una latenza aggiuntiva trascurabile e senza costi aggiuntivi. Questa funzionalità è inizialmente disponibile nelle seguenti regioni: Stati Uniti orientali (Virginia settentrionale), Stati Uniti occidentali (Oregon), Europa (Francoforte), Europa (Irlanda), Asia Pacifico (Sydney), Asia Pacifico (Tokyo), Asia Pacifico (Seoul). Per ulteriori informazioni sulle sedi di assistenza, vedere. [Amazon GameLift Serverssedi di assistenza](#)

Ulteriori informazioni:

- [Post sul blog: Presentazione della protezione Amazon GameLift Servers DDo S](#)
- [DDoProtezione S con Amazon GameLift Servers](#), Guida Amazon GameLift Servers per gli sviluppatori
- [Come funziona Player Gateway](#), Guida Amazon GameLift Servers per gli sviluppatori
- [Integra il gateway del giocatore in un gioco](#), Guida Amazon GameLift Servers per gli sviluppatori
- [Client SDK per C++](#)
- [Client SDK](#) per Unreal. L'accesso richiede l'iscrizione all'organizzazione Epic Games il. GitHub Vedi [Unreal Engine su GitHub per maggiori dettagli](#).

24 febbraio 2026: i Amazon GameLift Servers plugin ora supportano Unity 6.3 e Unreal Engine 5.7

Versioni aggiornate dei plugin:

- [Plugin per Unity](#), versione 3.2.1 (include il server SDK per Unity, versione 5.4.0)
- [Plugin per Unreal](#), versione 3.1.1 (include l'SDK del server per Unreal, versione 5.4.0)

Amazon GameLift Servers rilascia nuove versioni del plugin e del server SDK per Unreal Engine e Unity. Queste ultime versioni includono i seguenti aggiornamenti:

- Ora supportano l'uso con Unreal Engine 5.7 e Unity 6.3.

Ulteriori informazioni:

- [Integrazione dei giochi con il Amazon GameLift Servers plug-in per Unreal Engine, Guida per gli sviluppatori Amazon GameLift Servers](#)
- [Integrazione Amazon GameLift Servers in un progetto Unreal Engine](#), Guida per sviluppatori Amazon GameLift Servers
- [Integrazione dei giochi con il Amazon GameLift Servers plugin per Unity, Amazon GameLift Servers Guida per gli sviluppatori](#)
- [Integrazione Amazon GameLift Servers in un progetto Unity Engine](#), Guida per Amazon GameLift Servers sviluppatori

29 gennaio 2026: Amazon GameLift Servers ora supporta la scalabilità a zero istanze di Fleets to/from

Amazon GameLift Servers ora supporta il ridimensionamento automatico da e verso zero istanze in base all'attività della sessione di gioco. Le flotte si riducono a zero dopo un determinato periodo di assenza di attività di gioco e aumentano da zero quando vengono richieste sessioni di gioco, offrendo un'opzione per l'ottimizzazione dei costi.

Amazon GameLift Servers Link di riferimento API:

- [AWS Azione SDK UpdateFleetCapacity](#)

Ulteriori informazioni:

- [Gestisci la scalabilità di una flotta di GameLift server Amazon Zero To/From](#)

22 gennaio 2026: Amazon GameLift Servers Realtime ora supporta Node.js 24.x su Amazon Linux 2023

Amazon GameLift Servers Realtime ora supporta Node.js 24.x su Amazon Linux 2023. Questo aggiornamento fornisce l'accesso alle funzionalità e ai miglioramenti di sicurezza più recenti di Node.js durante l'esecuzione sul moderno sistema operativo Amazon Linux 2023.

Ora puoi anche includere uno script di installazione opzionale durante la creazione di una Realtime flotta, che consente l'installazione del software all'avvio dell'istanza per le tue esigenze specifiche. Per utilizzare la nuova versione di runtime, specifica Node.js 24.x durante la creazione Realtime degli script.

Amazon GameLift Servers Link di riferimento API:

- [AWS Azione SDK CreateScript](#)

Ulteriori informazioni:

- [Versioni di runtime di Node.js](#)
- [Aggiungere uno script di installazione](#)

9 dicembre 2025: Amazon GameLift Servers ora fornisce assistenza basata sull'intelligenza artificiale nella console web

Amazon GameLift Servers ora fornisce assistenza basata sull'intelligenza artificiale nella console Web, sfruttando Amazon Q Developer per fornire indicazioni personalizzate agli sviluppatori di giochi. Questa nuova funzionalità integra Amazon GameLift Servers conoscenze specialistiche per aiutare i clienti a navigare in flussi di lavoro complessi, risolvere problemi e ottimizzare le implementazioni dei server di gioco in modo più efficiente.

Questa nuova funzionalità fornisce consigli basati sull'intelligenza artificiale per l'integrazione dei server di gioco, la configurazione della flotta e l'ottimizzazione delle prestazioni direttamente all'interno della console. Amazon GameLift Servers Questo miglioramento mira a semplificare i processi decisionali, ridurre i tempi di risoluzione dei problemi e migliorare l'utilizzo complessivo delle risorse, con conseguenti risparmi sui costi e migliori esperienze per i giocatori.

Ulteriori informazioni:

- [Console Amazon GameLift Servers](#)

6 novembre 2025: Amazon GameLift Servers ora supporta le build di gioco che utilizzano il sistema operativo Windows Server 2022

Amazon GameLift Servers i clienti possono ora utilizzare il sistema operativo Windows Server 2022 per ospitare i propri server di gioco. Windows Server 2022 offre diversi miglioramenti rispetto a Windows Server 2016, inclusa la sicurezza. Questo sistema operativo è disponibile in tutte le regioni AWS ad eccezione delle regioni della Cina.

I clienti possono utilizzare i più recenti sistemi operativi Windows e continuare a ricevere aggiornamenti di sicurezza critici quando il supporto per Windows Server 2016 terminerà a gennaio 2027. Il supporto per Windows Server 2022 proseguirà fino al 13 ottobre 2031.

Amazon GameLift Servers Link di riferimento API:

- [AWS Azione SDK CreateBuild](#)
- [Comando CLI upload-build](#)
- [Comando CLI create-build](#)

28 ottobre 2025: Amazon GameLift Servers aggiunge il raccoglitore di metriche OpenTelemetry (OTel) a tutti i server e i plugin SDKs

Amazon GameLift Servers ora include la raccolta di metriche integrata OpenTelemetry (OTel) in tutti i server SDKs e i plugin, fornendo un'osservabilità standardizzata per il monitoraggio delle prestazioni dei server di gioco. Questa integrazione offre funzionalità di telemetria standard del settore direttamente sui server di gioco, consentendo un monitoraggio completo e la risoluzione dei problemi senza richiedere strumentazione o configurazione aggiuntive.

Con OpenTelemetry l'integrazione, ora puoi:

- Acquisisci automaticamente le metriche, i log e le tracce essenziali dei server di gioco utilizzando protocolli standardizzati
- Si integra perfettamente con le piattaforme di monitoraggio e gli strumenti di osservabilità esistenti
- Monitora le prestazioni e lo stato dei server di gioco in tempo reale su tutte le istanze della flotta
- Risolvi i problemi più rapidamente con i dati di telemetria che funzionano su diverse soluzioni di monitoraggio

OpenTelemetry è un framework di osservabilità open source che fornisce un modo standardizzato per raccogliere, elaborare ed esportare dati di telemetria.

Ulteriori informazioni:

- [Monitoraggio con metriche di telemetria del server](#) Amazon GameLift Servers, Guida per gli sviluppatori

2 ottobre 2025: Amazon GameLift Servers aggiunge l'accesso remoto basato su console alle istanze del parco istanze

Amazon GameLift Servers ora fornisce l'accesso remoto diretto alle istanze gestite di EC2 e di flotte di container tramite la Amazon GameLift Servers console, eliminando la necessità di gestire l'autenticazione. Questa nuova funzionalità utilizza Amazon EC2 Systems Manager (SSM) per fornire un accesso sicuro ai terminali basato su browser senza richiedere configurazioni o gestione delle credenziali aggiuntive.

Con l'accesso remoto basato su console, ora puoi:

- Risolvi i problemi di integrazione dei server di gioco in tempo reale
- Monitora l'attività del server di gioco e accedi ai file di registro durante le sessioni di gioco attive
- Esegui strumenti di diagnostica e benchmarking utilizzando il traffico effettivo dei giocatori

Questa funzionalità è disponibile per le flotte che eseguono la versione 5.x del server SDK. Per le flotte che eseguono versioni SDK precedenti, puoi continuare a utilizzare il metodo AWS CLI esistente per l'accesso remoto.

Per connetterti in remoto a un'istanza della flotta, apri la console Amazon GameLift Servers e trova l'EC2 o la flotta di container gestita a cui desideri accedere. Nei dettagli del parco istanze, apri la scheda Istanze, seleziona un'istanza dall'elenco e scegli Connect.

Ulteriori informazioni:

- [Connect a flotte di istanze](#), Guida per Amazon GameLift Servers agli sviluppatori

23 settembre 2025: Amazon GameLift Servers aggiunge il supporto per la nuova zona locale a Dallas, Texas

Con l'hosting Amazon GameLift Servers gestito, ora puoi distribuire le risorse del server di gioco in una nuova zona locale di Dallas, TX (`us-east-1-dfw-2`). Le Local Zones ti danno la possibilità di posizionare i server di gioco geograficamente più vicini ai giocatori per ridurre la latenza e migliorare significativamente il gameplay. In questa nuova zona locale, puoi utilizzare i seguenti tipi di istanze EC2: C6gn, C6i, C6in, M6g, M6i, M6in, M8g e R6i. Per una descrizione dettagliata [AWS di ogni tipo di istanza](#), consulta [Tipi di istanza EC2](#).

Per iniziare a ospitare sessioni di gioco nella nuova zona locale di Dallas, accedi innanzitutto alla zona locale della tua. Account AWS Quindi puoi aggiungerla come postazione remota a una flotta con più sedi nuova o esistente. Se il tuo gioco funziona Amazon GameLift ServersFlexMatch, aggiorna una delle flotte presenti nella tua coda di matchmaking per includere la nuova Zona locale. Con le flotte con più sedi, puoi gestire direttamente la capacità di hosting in ciascuna località.

Il Regione AWS capostipite della nuova zona locale di Dallas, TX, è `us-east-1` (Virginia). La versione originale di Dallas, TX, Local Zone (`us-east-1-dfw-1`) non accetta più nuove richieste di opt-in.

Ulteriori informazioni:

- [Guida introduttiva](#), [Guida per l'utente Zone locali AWS](#) [Zone locali AWS](#)
- [Amazon GameLift Servers](#) [sedi di assistenza](#), Guida Amazon GameLift Servers per gli sviluppatori
- [Aggiorna le posizioni della flotta](#), Guida Amazon GameLift Servers per gli sviluppatori

12 agosto 2025: il Amazon GameLift Servers plugin per Unreal Engine ora supporta la versione UE 5.6

Versione aggiornata del plugin:

- [Plugin per Unreal](#), versione 3.0.1 (include l'SDK del server per Unreal, versione 5.3.1)

Amazon GameLift Serversrilascia nuove versioni del plugin per Unreal Engine e del server SDK per Unreal Engine. Queste ultime versioni includono i seguenti aggiornamenti:

- Ora supportano l'uso con Unreal Engine 5.6.
- I problemi di compilazione durante il pacchetto degli obiettivi dei client Android sono stati risolti.

Ulteriori informazioni:

- [Integrazione dei giochi con il Amazon GameLift Servers plug-in per Unreal Engine, Guida per gli sviluppatori Amazon GameLift Servers](#)
- [Integrazione Amazon GameLift Servers in un progetto Unreal Engine](#), Guida per sviluppatori Amazon GameLift Servers

24 giugno 2025: Amazon GameLift Servers lancia i fari di ping UDP per aiutare a misurare la latenza di rete in tempo reale per i giochi multiplayer

Versioni SDK aggiornate:

- SDK AWS 1.11.595

Amazon GameLift Servers rilascia per la disponibilità generale una serie di endpoint fissi chiamati ping beacon UDP per aiutarti a misurare con precisione la latenza tra i dispositivi dei giocatori e le posizioni dei server di gioco. Gli endpoint dei beacon ping UDP sono disponibili in tutte le regioni AWS globali e le Local Zones supportate da Amazon GameLift Servers, ad eccezione delle regioni della Cina. AWS

La maggior parte dei giochi multiplayer utilizza UDP (User Datagram Protocol) come protocollo di trasmissione di pacchetti principale, grazie ai vantaggi in termini di prestazioni per i giochi in tempo reale. Comprendere e ottimizzare la latenza di rete è fondamentale per offrire la migliore esperienza di gioco possibile. I ping beacon UDP forniscono un modo coerente e affidabile per misurare la latenza effettiva dei pacchetti UDP tra giocatori e server di gioco, aiutandoti a prendere decisioni migliori sugli abbinamenti e sul posizionamento delle sessioni di gioco. player-to-server

L'[ListLocations](#) API è stata estesa per includere le informazioni sul dominio e sulla porta degli endpoint come parte dell'elenco delle posizioni restituite, semplificando l'accesso programmatico agli endpoint.

Il tuo client di gioco può inviare messaggi UDP a questi endpoint e ricevere risposte asincrone con gli stessi dati, fornendoti misurazioni della latenza che rappresentano meglio le condizioni effettive del traffico di gioco tra il dispositivo di un giocatore e le potenziali località di hosting. Questi endpoint sono permanenti e rimangono disponibili fintanto che supporta l'hosting di giochi in quella località. Amazon GameLift Servers

Ulteriori informazioni:

- [UDP ping beacon](#), Guida per Amazon GameLift Servers agli sviluppatori
- [ListLocations](#), Riferimento Amazon GameLift Servers API

24 giugno 2025: Amazon GameLift Servers aggiunge il supporto di localizzazione per Thailandia e Malesia

Con l'hosting Amazon GameLift Servers gestito, ora puoi distribuire risorse di server di gioco a Bangkok, in Thailandia, e Kuala Lumpur, in Malesia, estendendo la portata dei tuoi giochi ai giocatori di tutto il sud-est asiatico. Queste nuove regioni aiutano a ridurre la latenza e a migliorare l'esperienza di gioco dei giocatori in quelle aree.

Le seguenti Regioni AWS sono disponibili come postazioni remote per flotte con più sedi. Per iniziare a ospitare sessioni di gioco in queste località, aggiungile come postazioni remote a una flotta con più sedi nuova o esistente. Con le flotte con più sedi, puoi gestire direttamente la capacità di hosting in ogni sede.

- Asia Pacifico (Thailandia) `ap-southeast-7` ()
- Asia Pacifico (Malesia) () `ap-southeast-5`

Questi non Regioni AWS sono abilitati per impostazione predefinita per un AWS account. È necessario iscriversi a ciascuna regione prima di potervi distribuire Amazon GameLift Servers risorse.

Ulteriori informazioni:

- [Amazon GameLift Servers sedi di assistenza](#), Guida per sviluppatori Amazon GameLift Servers di hosting
- [Aggiorna le posizioni della flotta](#), Guida per sviluppatori di Amazon GameLift Servers hosting
- [Abilita o disabilita AWS le regioni nel tuo AWS account](#), Guida di riferimento per la gestione degli account.

29 maggio 2025: Amazon GameLift Servers rilascia nuove versioni dell'SDK del server e dei plugin per Unreal Engine e Unity

Versioni SDK aggiornate:

- [SDK del server C++](#), versione 5.3.0

- SDK per [server C#](#), versione 5.3.0
- [Go server SDK](#), versione 5.3.0
- [Server SDK per Unreal](#), versione 5.3.0
- [Server SDK per Unity](#), versione 5.3.0

Versioni aggiornate del plugin:

- [Plugin per Unreal](#), versione 3.0.0
- [Plugin per Unity](#), versione 3.1.0

Le nuove versioni del server SDK per C++, C#, Go, Unreal e Unity, nonché le nuove versioni dei plugin per Unreal Engine e Unity, sono ora open source. [Sono tutte disponibili nell'organizzazione. Amazon GameLift Servers GitHub](#). Per informazioni dettagliate sugli aggiornamenti, consulta le note di rilascio e i readme in ogni repository.

Aggiornamenti chiave dell'SDK del server:

- Convalida lato client e risposte agli errori migliorate in tutti i server. SDKs
- La funzione di [OnProcessTerminate](#) callback ora ha una logica predefinita per terminare il processo del server di gioco.
- La funzione [InitSDK\(\)](#) ora utilizza un token di idempotenza per supportare più tentativi.
- Il [OnUpdateGameSession](#) callback può ora passare dati per. `autoBackfillMode`

Aggiornamenti chiave del plugin:

- Il plugin per Unreal Engine ora ha un processo di installazione e configurazione più semplificato, con maggiore automazione e meno prerequisiti (CMakeOpenSSL e la catena di strumenti di compilazione incrociata Unreal).
- Il plug-in per Unreal Engine ha migliorato l'esperienza dell'interfaccia utente per il flusso di lavoro Managed EC2, incluso il supporto per gli spazi nei percorsi di creazione del client di gioco e del server. Inoltre, ora puoi aggiungere argomenti da riga di comando quando avvii un client di gioco dall'Editor.
- Il plugin per Unreal Engine ora supporta le build del server ARM. UE5

Ulteriori informazioni:

- [Amazon GameLift Serversplugin per Unreal Engine](#), Amazon GameLift Servers Guida per gli sviluppatori
- [Amazon GameLift Serversplugin per Unity \(server SDK 5.x\)](#), Guida Amazon GameLift Servers per gli sviluppatori
- [Server SDK 5.x per Amazon GameLift Servers](#), Guida Amazon GameLift Servers per gli sviluppatori
- [Integrazione Amazon GameLift Servers in un progetto Unreal Engine](#), Guida Amazon GameLift Servers per gli sviluppatori
- [Integrazione Amazon GameLift Servers in un progetto Unity](#), Guida Amazon GameLift Servers per gli sviluppatori

24 aprile 2025: il server C# SDK 5 per Amazon GameLift Servers ora supporta .NET 8

Versioni SDK aggiornate:

- C# Server SDK, versione 5.2.1

Per gli sviluppatori di giochi che utilizzano C#, ora puoi utilizzare .NET 8 come framework di destinazione per i tuoi progetti. Amazon GameLift Servers Con .NET 8, puoi sfruttare i miglioramenti delle prestazioni, tra cui una migliore compilazione just-in-time (JIT), l'ottimizzazione dell'utilizzo della memoria e tempi di avvio più rapidi. Se attualmente utilizzi .NET 6, ti consigliamo di pianificare una migrazione a .NET 8, incluso l'aggiornamento dell'SDK del server C# alla versione più recente. Microsoft ha annunciato il supporto di .NET 8, con patch di sicurezza e aggiornamenti tecnici, fino a novembre 2026.

[Scarica la versione più recente dell'SDK del server C# per Amazon GameLift Servers](#)

Ulteriori informazioni:

- [Ottieni strumenti Amazon GameLift Servers di sviluppo](#), Amazon GameLift Servers Guida per gli sviluppatori

27 marzo 2025: Amazon GameLift Servers amplia il supporto per le famiglie di istanze EC2 di nuova generazione

Versioni SDK aggiornate:

- AWS SDK 1.11.535

Ora puoi ottimizzare l'hosting del tuo server di gioco Amazon GameLift Servers scegliendo tra una selezione più ampia di istanze Amazon EC2 per famiglie di istanze di quinta e ottava generazione. Ogni nuova generazione EC2 offre miglioramenti nell'elaborazione, nella memoria e nel networking EC2, con le istanze di ottava generazione che offrono istanze all'avanguardia basate su Graviton4 e Intel Xeon. AWS Le istanze di nuova generazione sono disponibili nelle seguenti famiglie di istanze:

- [Uso generico](#) (serie M)
- [Ottimizzato per il calcolo \(serie C\)](#)
- [Memoria ottimizzata](#) (serie R)

È inoltre possibile scegliere varianti che offrono storage locale (d), rete avanzata (n) e architetture di processore specifiche (). Intel/AMD/Graviton – i/a/g Le istanze di nuova generazione sono disponibili in tutte Regioni AWS le aree supportate da Amazon GameLift Servers, ad eccezione delle regioni della Cina AWS . Per ulteriori dettagli, consultare [Amazon GameLift Servers sedi di assistenza](#).

Utilizza questi nuovi tipi di istanze con flotte EC2 Amazon GameLift Servers gestite e flotte di container gestite. Quando passi il tuo hosting di giochi esistente a un nuovo tipo di istanza (stessa architettura), è sufficiente implementare nuove flotte con tutte le impostazioni di configurazione invariate tranne il tipo di istanza.

Ulteriori informazioni:

- [Amazon GameLift Servers Prezzi delle istanze](#)
- [CreateFleet Amazon GameLift Servers](#) Riferimento alle API
- [CreateContainerFleet](#), Riferimento Amazon GameLift Servers API

11 marzo 2025: Amazon GameLift Servers lancia il nuovo Game Server Wrapper per un rapido onboarding

Il nuovo game server wrapper riduce Amazon GameLift Servers significativamente il tempo necessario per ospitare un server di gioco su cui è ospitato. Amazon GameLift Servers Senza dover modificare il codice, puoi utilizzare il wrapper per aggiungere funzionalità di base per la gestione delle sessioni di gioco al tuo gioco e distribuirlo su una flotta Amazon GameLift Servers Anywhere, una flotta EC2 gestita o una flotta di container gestita. Questo strumento è ideale per effettuare una

valutazione pratica delle Amazon GameLift Servers funzionalità, utilizzando il proprio server di gioco o uno tratto da un gioco di esempio. È utile anche per implementare rapidamente iterazioni di server di gioco, ad esempio per prototipazione o test rapidi.

Grazie alle funzionalità di base per la gestione delle sessioni di gioco, il server di gioco può inizializzare una connessione con il Amazon GameLift Servers servizio, rispondere alle richieste di avvio e interruzione delle sessioni di gioco e spegnersi al termine di una sessione di gioco.

[Scarica il wrapper del server di gioco da GitHub](#) Consulta il readme su come installare e utilizzare il GitHub wrapper con i tuoi progetti di gioco.

6 marzo 2025: Amazon GameLift è ora Amazon GameLift Servers e Amazon Streams GameLift

Amazon GameLift è un servizio completamente gestito dedicato ad aiutare gli sviluppatori a creare, scalare e fornire i giochi più impegnativi al mondo. Con la versione di disponibilità generale di Amazon GameLift Streams, Amazon offre GameLift ora sia server di gioco ad alta scalabilità che funzionalità di streaming di gioco fluide.

[Amazon GameLift Servers](#) offre agli sviluppatori di giochi la possibilità di implementare, gestire e scalare server di gioco dedicati. Puoi implementare server di gioco ad alte prestazioni nel cloud in pochi minuti e scalare verso l'alto e verso il basso per soddisfare le esigenze dei giocatori. Basato su un ambiente informatico AWS collaudato, Amazon GameLift Servers supporta 100 milioni di giocatori simultanei in una singola partita, 100.000 aggiunte di giocatori al secondo e 9.000 nuove istanze di calcolo al minuto. Inoltre, grazie alla sicurezza di livello aziendale, al matchmaking per il pubblico più numeroso e alla pay-as-you-go flessibilità, ti aiuta a iniziare sia che tu stia lavorando a una nuova idea di gioco sia che tu stia gestendo una partita con milioni di giocatori.

[Amazon GameLift Streams](#) aiuta gli sviluppatori di giochi a offrire esperienze di streaming di gioco con una risoluzione fino a 1080p e 60 frames-per-second (fps) senza ritardi percepibili su dispositivi tra cui iOS, Android, FireOS e per i giocatori. PCs Utilizzando un'unica AWS offerta, gli editori possono distribuire i propri contenuti di gioco in pochi minuti, senza modifiche, su istanze GPU basate su cloud completamente gestite e distribuirli tramite AWS Network Backbone direttamente a qualsiasi dispositivo del giocatore finale dotato di browser web. I giocatori possono iniziare a giocare in pochi secondi, senza attendere il download o l'installazione e offre un'esperienza di gioco quasi indistinguibile dal gioco in locale su un PC o una console di gioco.

14 gennaio 2025: Amazon GameLift Servers rilascia il supporto esteso per la prioritizzazione del posizionamento delle sessioni di gioco

Versioni SDK aggiornate:

- AWS SDK 1.11.485

In risposta al feedback dei clienti, stiamo rilasciando una nuova funzionalità che consente di assegnare priorità alle posizioni per le richieste di posizionamento delle singole sessioni di gioco. Per le code configurate per dare priorità al posizionamento in base alla località, ora puoi fornire un elenco personalizzato di posizioni prioritarie con ogni richiesta di posizionamento.

Questa nuova funzionalità consente ai clienti di modificare dinamicamente le priorità di localizzazione per ogni richiesta di collocamento in base alle esigenze. La flessibilità aggiuntiva consente di rispondere meglio alle mutevoli condizioni, come la posizione dei giocatori, il carico della flotta o lo stato del server. Può anche supportare i clienti che desiderano personalizzare ulteriormente il modo in cui vengono selezionate le sedi di collocamento.

Ulteriori informazioni:

- [Dai priorità al posizionamento delle sessioni di gioco](#), Amazon GameLift Servers Guida per gli sviluppatori
- [StartGameSessionPlacement](#), Amazon GameLift Servers Riferimento alle API

2 gennaio 2025: Amazon GameLift Servers rilascia un nuovo supporto per l'interruzione delle sessioni di gioco

Versioni SDK aggiornate:

- AWS SDK 1.11.477

In risposta al feedback dei clienti, stiamo rilasciando nuove funzionalità che consentono di terminare più facilmente le singole sessioni di gioco. Con questa versione, ora puoi terminare una sessione di gioco direttamente nella Amazon GameLift Servers console o utilizzando l'SDK AWS CLI o AWS per. Amazon GameLift Servers

Questa nuova funzionalità risponde alla necessità di risolvere sessioni di gioco che rimangono attive ma in cattivo stato, impedendo così alle risorse di calcolo di ospitare nuove sessioni di gioco. In

precedenza, i clienti dovevano accedere in remoto al computer per terminare manualmente una sessione di gioco.

Hai due metodi di terminazione tra cui scegliere. Il primo metodo tenta di terminare correttamente una sessione di gioco utilizzando la sequenza di spegnimento personalizzata, che potrebbe includere azioni per avvisare i giocatori e risolvere i dati di gioco. Il secondo metodo impone l'interruzione del processo del server, che interrompe immediatamente la sessione di gioco. Questo secondo metodo assicura che la sessione di gioco termini anche quando il processo del server non risponde.

Ulteriori informazioni:

- [Chiudi una sessione di gioco utilizzando la Amazon GameLift Servers console, Guida](#) per Amazon GameLift Servers agli sviluppatori
- [TerminateGameSession](#), Riferimento Amazon GameLift Servers API

19 dicembre 2024: Amazon GameLift Servers rilascia il supporto per i plugin del motore di gioco per Managed Containers

Versioni aggiornate del plugin:

Amazon GameLift Servers plugin per Unreal Engine, versione 2.0.0

- Aggiornato per supportare il server C++ SDK 5.2.0 con supporto per contenitori gestiti.
- È stato aggiunto il supporto per Unreal Engine 5.4 e 5.5.

Amazon GameLift Servers plugin per Unity, versione 3.0.0

- Aggiornato per supportare il server C++ SDK 5.2.0 con supporto per contenitori gestiti.
- Support per Unity 2021.3 LTS e 2022.3 LTS per Windows e Mac OS.

Il Amazon GameLift Servers plug-in per i motori di gioco Unreal e Unity fornisce strumenti e flussi di lavoro che semplificano i passaggi per avviare e utilizzare un gioco. Amazon GameLift Servers è un servizio di hosting cloud completamente gestito che gli sviluppatori di giochi possono utilizzare per gestire e scalare server di gioco dedicati per giochi multiplayer basati su sessioni.

Le ultime versioni dei plugin offrono i seguenti miglioramenti:

- Flusso di lavoro guidato per l'hosting con Managed Containers. Questo flusso di lavoro illustra i passaggi per configurare un'immagine del contenitore con il software del server di gioco e implementare una soluzione di hosting basata su cloud per il server di gioco. Il flusso di lavoro offre due diversi scenari di implementazione: una distribuzione semplice e una distribuzione più completa con una coda per il posizionamento delle sessioni di gioco e un matchmaker. FlexMatch Ogni scenario genera flotte di Amazon GameLift Servers container e risorse di supporto. AWS
- Processo migliorato per la configurazione dei profili AWS utente e la gestione delle credenziali di AWS accesso per l'uso dei plug-in. Puoi gestire più profili per lavorare con AWS account, utenti di account e aree geografiche diversi.
- Funzionalità aggiuntive per aggiornare le flotte di container esistenti. Puoi distribuire nuove immagini di container (ad esempio per gli aggiornamenti delle versioni dei server di gioco) e modificare le impostazioni di configurazione della flotta senza dover ricominciare dall'inizio.
- Flussi di lavoro migliorati per l'hosting con flotte Amazon GameLift Servers Anywhere e Managed EC2. I miglioramenti basati sul feedback dei clienti includono una guida migliore con suggerimenti e link a risorse utili.

Gli scenari di implementazione per i container gestiti e le soluzioni Managed EC2 utilizzano AWS CloudFormation modelli per creare e distribuire le AWS risorse per ogni scenario. Questi modelli sono inclusi nel download del Amazon GameLift Servers plug-in e sono modificabili. Puoi usarli così come sono o modificarli per il tuo gioco.

Ulteriori informazioni:

- [Plugin per Unreal: distribuisce il tuo gioco su una flotta di container gestita, Amazon GameLift Servers Guida](#) per sviluppatori
- [Plugin per Unity: distribuisce il gioco su una flotta di container gestita](#), Guida per sviluppatori Amazon GameLift Servers
- [Scarica il plugin per Unreal da GitHub](#)
- [Scarica il plugin per Unity da GitHub](#)
- [Hosting di giochi con Amazon GameLift Servers](#)
- [Forum Amazon GameLift Servers](#)

12 novembre 2024: Amazon GameLift Servers lancia l'hosting di giochi multiplayer con Managed Containers

Versioni SDK aggiornate:

- AWS SDK 1.11.445
- Server SDK, versione 5.2.0 (tutte le lingue)

Amazon GameLift Servers rilascia per la disponibilità generale una nuova soluzione di hosting per carichi di lavoro su server di gioco containerizzati. Con questa versione, gli sviluppatori di giochi possono ora sfruttare i vantaggi della containerizzazione, tra cui ambienti coerenti e sicuri, un processo di distribuzione semplificato e un utilizzo ottimizzato delle risorse.

Le flotte di container gestite utilizzano istanze Amazon EC2 Amazon GameLift Servers gestite per tuo conto e in base alle tue configurazioni. Puoi creare un'architettura di container personalizzata per il tuo gioco e fornire immagini di container archiviandole in un repository Amazon Elastic Container Registry (Amazon ECR). Le flotte di container sono disponibili solo per i server di gioco basati su Linux. I server di gioco devono essere integrati con Server SDK 5.2.0 o versione successiva.

Con le flotte di container gestite, ottieni gli stessi vantaggi delle flotte EC2 gestite. Ciò include il supporto per i tipi di istanze On-Demand e Spot, la scalabilità intelligente della capacità, il posizionamento delle sessioni di gioco con code e il matchmaking. Inoltre, ottieni le stesse metriche degli altri tipi di flotta, oltre ad alcune nuove metriche per i container. Altre caratteristiche per le flotte di container includono:

- Allineamento con l'esperienza serverless per carichi di lavoro containerizzati. Esegui un processo del server di gioco per container e raccogli molti container in ogni istanza del parco macchine per un utilizzo ottimale delle risorse. Se preferisci avere contenitori con più processi del server di gioco, puoi utilizzare l'Amazon GameLift Servers agente per la gestione automatizzata degli host.
- Creazione semplificata della flotta. Le flotte di container sono progettate per richiedere impostazioni di configurazione di implementazione minime, con valori suggested/default ragionevoli. È possibile implementare rapidamente una flotta funzionante e quindi personalizzare le impostazioni individuali in base alle esigenze.
- Strumenti di controllo delle versioni per l'architettura dei container. Ora puoi aggiornare la definizione di un gruppo di container (che è simile a un «task» di container), mantenere più versioni e specificare quale versione distribuire in una flotta.

- Strumenti di aggiornamento della flotta. Con le flotte di container, non è più necessario creare una nuova flotta quando si desidera rilasciare un aggiornamento della versione del server di gioco. Invece, ora puoi aggiornare l'immagine del container e distribuire gli aggiornamenti alle flotte esistenti.

Puoi creare flotte di Amazon GameLift Servers container Regione AWS ovunque Amazon GameLift Servers supporti flotte con più sedi e distribuire istanze di flotte di container in qualsiasi posizione remota supportata. Per ulteriori dettagli, consultare [Amazon GameLift Servers sedi di assistenza](#). I container gestiti non sono attualmente disponibili nelle regioni AWS della Cina.

Ulteriori informazioni:

- Post del blog: [Sfrutta i container completamente gestiti per ospitare partite multiplayer su scala globale su Amazon GameLift Servers](#)
- [Contenitori gestiti](#) panoramica, guida per Amazon GameLift Servers per gli sviluppatori
- [Come funzionano i contenitori Amazon GameLift Servers](#), Guida Amazon GameLift Servers per gli sviluppatori
- [Roadmap di sviluppo per l'hosting con contenitori Amazon GameLift Servers gestiti](#), Guida Amazon GameLift Servers per gli sviluppatori
- [CreateContainerFleet](#), Riferimento Amazon GameLift Servers API

19 settembre 2024: Amazon GameLift Servers rilascia l'aggiornamento dell'SDK del server C++ e dei plugin per Unreal Engine

Versioni SDK aggiornate:

C++ Server SDK, versione 5.1.3

- Nuove funzionalità di registrazione. Ora puoi accedere ai registri delle richieste SDK.
- Migliore affidabilità della trasmissione dei messaggi SDK. L'SDK ora utilizza meccanismi di riconnessione più robusti per il ripristino in caso di interruzioni di rete o cadute casuali dei messaggi.

Versioni aggiornate del plugin:

Amazon GameLift Servers plugin per Unreal Engine, versione 1.1.2

- Aggiornato per supportare l'ultima versione del server C++ SDK 5.1.3.
- Nel Amazon GameLift Servers plugin per Unreal Engine, quando cerchi un eseguibile di build del server per una flotta, ora hai la possibilità di sfogliare Tutti i file.

Plugin SDK C++ Server per Unreal, versione 5.1.2

- Aggiornato per supportare l'ultima versione del server C++ SDK 5.1.3.

Ulteriori informazioni:

- [Integrazione dei giochi con il Amazon GameLift Servers plug-in per Unreal Engine, Guida per gli sviluppatori Amazon GameLift Servers](#)
- [Amazon GameLift Serversdownload di plugin e SDK](#)

5 settembre 2024: Amazon GameLift Servers migliora l'osservabilità del processo di creazione della flotta

Sulla base del feedback dei clienti, abbiamo chiarito il Amazon GameLift Servers flusso di lavoro per creare una flotta EC2 gestita e prepararla per ospitare sessioni di gioco. Miglioramenti:

- Abbiamo fornito descrizioni più specifiche e accurate di ogni fase del processo di creazione della flotta. Questa visibilità migliorata semplifica l'individuazione e la risoluzione dei problemi più rapidamente.
- Le fasi di creazione e attivazione separano meglio le attività di distribuzione delle istanze (creazione) dalle attività per avviare i processi del server di gioco e connettersi al Amazon GameLift Servers servizio (attivazione). Questa modifica semplifica il riconoscimento della probabile causa dei problemi. Inoltre, ora puoi connetterti in remoto alle flotte quando sono in fase di attivazione.
- Due nuovi eventi di creazione della flotta comunicano l'esito positivo o negativo degli script di installazione dei server di gioco. Se la build del server di gioco include uno script di installazione, Amazon GameLift Servers tenta di eseguire lo script ed emette uno dei seguenti nuovi eventi:
 - FLEET_CREATION_COMPLETED_INSTALLER
 - FLEET_CREATION_FAILED_INSTALLER

Ulteriori informazioni:

- [Debug dei problemi del parco istanze di Amazon GameLift Servers](#), Guida per Amazon GameLift Servers agli sviluppatori
- [Tipo di dati dell'evento](#), riferimento Amazon GameLift Servers API

15 agosto 2024: Amazon GameLift Servers migliora l'esperienza della console

In base al feedback dei clienti, abbiamo apportato i seguenti aggiornamenti all'esperienza della [Amazon GameLift Servers console](#):

- Le tue preferenze di visualizzazione per le pagine vengono ora salvate automaticamente nell' AWS account utente e applicate ogni volta che torni alla pagina. Le preferenze di visualizzazione consentono di scegliere quali informazioni includere in una tabella, ad esempio nella pagina di elenco Fleets. Personalizza le tue preferenze di visualizzazione utilizzando l' icona nell'angolo in alto a destra di una tabella.
- Il flusso di lavoro Create Fleet per flotte EC2 gestite è stato semplificato per combinare la selezione di sedi del parco veicoli e tipi di istanze. Abbiamo semplificato la ricerca del tipo di istanza giusto per la tua flotta, anche quando modifichi le selezioni delle sedi.

Ulteriori informazioni:

- [Crea una EC2 flotta Amazon GameLift Servers gestita](#), Guida per Amazon GameLift Servers agli sviluppatori

25 luglio 2024: Amazon GameLift Servers aggiunge il supporto per la zona locale AWS della Nigeria

Con l'hosting Amazon GameLift Servers gestito, ora puoi distribuire risorse di server di gioco in Nigeria, Africa occidentale ed estendere la portata dei tuoi giochi ai giocatori di tutta l'Africa. Usa AWS Local Zones per posizionare i server di gioco geograficamente più vicini ai tuoi giocatori per ridurre la latenza e migliorare significativamente il gameplay.

Per iniziare subito a ospitare sessioni di gioco in Nigeria, aggiungi la nuova Zona locale della Nigeria come postazione remota a una flotta multisede nuova o esistente. Se il gioco lo utilizza Amazon

GameLift Servers FlexMatch, aggiorna le flotte nella coda di matchmaking per includere la nuova zona locale. Con le flotte con più sedi, puoi gestire direttamente la capacità di hosting in ciascuna località.

L'origine della zona locale Regione AWS di Lagos, in Nigeria, è la regione Africa (Città del Capo) `af-south-1` (), Amazon GameLift Servers che funge anche da postazione remota. Il nome della zona locale della Nigeria è `af-south-1-los-1`

Ulteriori informazioni:

- [Amazon GameLift Servers sedi di assistenza](#), Guida per Amazon GameLift Servers agli sviluppatori
- [Aggiorna le posizioni della flotta](#), Guida Amazon GameLift Servers per gli sviluppatori

02 luglio 2024: Amazon GameLift Servers rilascia un nuovo strumento su console per visualizzare i dati della sessione dei giocatori

La Amazon GameLift Servers console offre ora uno strumento di ricerca della sessione giocatore che consente di recuperare le informazioni sulla sessione del giocatore in base all'ID della sessione di gioco, all'ID della sessione del giocatore o all'ID giocatore. I giochi che utilizzano il FlexMatch matchmaking generano automaticamente sessioni di gioco per ogni giocatore abbinato. Per tutti gli altri giochi, le sessioni dei giocatori sono una funzionalità opzionale.

Puoi trovare lo strumento di ricerca della sessione dei giocatori nella navigazione principale della console. Amazon GameLift Servers Visualizza le sessioni dei singoli giocatori o confronta i dati tra sessioni con più giocatori. Puoi anche aprire i dati della sessione del giocatore quando visualizzi la pagina dei dettagli di una sessione di gioco.

Ulteriori informazioni:

- [Sessioni di gioco e di gioco nella Amazon GameLift Servers console](#), Guida Amazon GameLift Servers per gli sviluppatori

24 aprile 2024: Amazon GameLift Servers rilascia l'anteprima delle flotte di container

Amazon GameLift Servers offre ora un'anteprima delle flotte di container, che offrono portabilità, scalabilità, tolleranza ai guasti e agilità migliorate.

Nelle flotte di container, le istanze Amazon EC2 ospitano uno o più container. Questi contenitori includono il tuo server di gioco e tutto ciò che richiede, comprese le dipendenze e le configurazioni.

Esempi di dipendenze includono pacchetti software SDKs . Dopo aver caricato il container nel tuo Amazon Elastic Container Registry privato, Amazon GameLift Servers popola la tua flotta con il container.

Per funzionare in una flotta di container, il server di gioco deve funzionare in Linux ed essere integrato con Server SDK 5.x. In una flotta di container, hai un controllo preciso delle risorse di hosting in modo da poter ottimizzare il consumo di risorse come unità CPU e memoria. Puoi anche ospitare più server di gioco in un container per ridurre l'uso di risorse.

In una flotta di container ottieni molti degli stessi vantaggi di altri tipi di flotte, come i tipi di istanze On-Demand, la scalabilità (automatica e manuale), le code e il matchmaking. Inoltre, ottieni le stesse metriche degli altri tipi di flotta, oltre ad alcune nuove metriche per i container. Le flotte di container offrono una copertura globale ai giocatori nelle seguenti aree geografiche:

- ap-northeast-1
- ap-northeast-2
- ap-southeast-2
- eu-central-1
- eu-west-1
- us-east-1
- us-west-2

Per raggiungere ancora più regioni e zone locali, crea flotte di container con più sedi.

Ulteriori informazioni:

- [Gestione dell'hosting con Amazon GameLift Servers contenitori, Guida per](#) gli sviluppatori Amazon GameLift Servers
- [CreateContainerGroupDefinition](#), Riferimento alle Amazon GameLift Servers API

13 febbraio 2024: Amazon GameLift Servers lancia miglioramenti e semplifica l'installazione del Amazon GameLift Servers plug-in per SDKs Unreal Engine

Versioni SDK aggiornate:

- Go Server SDK, versione 5.1.0

- C# Server SDK, versione 5.1.2
- C++ Server SDK, versione 5.1.2

Abbiamo apportato i seguenti miglioramenti:

- È stata migliorata l'affidabilità dell'SDK aggiungendo la riconnessione automatica in caso di interruzione della rete.
- [Go] Ora puoi chiamare `InitSDK()` con o senza i parametri del server. I server di gioco che funzionano su flotte EC2 Amazon GameLift Servers gestite leggono i parametri del server direttamente dalle variabili di ambiente. I server di gioco sulle flotte Amazon GameLift Servers Anywhere devono effettuare chiamate `InitSDK()` con i parametri del server.

Versioni aggiornate dei plugin:

- Amazon GameLift Serversplugin per Unreal Engine, versione 1.1.0
- Amazon GameLift Serversplugin per Unity, versione 2.1.0
- Plugin C++ Server SDK per Unreal, versione 5.1.1
- Plugin C# Server SDK per Unity, versione 5.1.2

Abbiamo apportato i seguenti miglioramenti:

- [Amazon GameLift Serversplugin per Unreal Engine] Ha aggiornato le istruzioni di installazione e semplificato la confezione. Questo plugin ora include l'ultima versione di C++ Server SDK per Unreal.
- Sono stati aggiornati i plugin per supportare la versione più recente del server SDK per Amazon GameLift Servers

Ulteriori informazioni:

- [Integrazione dei giochi con il Amazon GameLift Servers plug-in per Unreal Engine, Guida per gli sviluppatori Amazon GameLift Servers](#)
- [Amazon GameLift Serversdownload di plugin e SDK](#)

14 dicembre 2023: Amazon GameLift Servers aggiunge la possibilità di aggiornare le proprietà di gioco delle sessioni di gioco attive

Sei già stato in grado di impostare le proprietà del gioco durante la creazione di sessioni di gioco e di cercare nelle sessioni di gioco proprietà specifiche. Ora puoi anche aggiungere e aggiornare queste proprietà in una sessione di gioco attiva.

Ad esempio, i tuoi giocatori votano su una mappa su cui vogliono giocare. Il client di gioco chiama `UpdateGameSession` per modificare un `GameProperty` valore in `{"Key": "map", "Value": "jungle"}`. Il gioco implementa quindi la nuova mappa per i giocatori nella sessione di gioco.

Gli amministratori del gioco possono anche recuperare dati utili dalle proprietà del gioco utilizzando l'operazione `SearchGameSessions`. Ad esempio, gli amministratori possono elencare le sessioni di gioco che hanno un `Status` valore `ACTIVE` e questa proprietà di gioco: `{"Key": "map", "Value": "desert"}`

Ulteriori informazioni:

- [the section called “Integra la funzionalità del client di gioco”](#), Guida per Amazon GameLift Servers per gli sviluppatori
- [GameProperty](#), Riferimento Amazon GameLift Servers API
- [UpdateGameSession](#), Riferimento Amazon GameLift Servers API
- [SearchGameSessions](#), Riferimento Amazon GameLift Servers API

21 novembre 2023: Amazon GameLift Servers lancia il supporto per strumenti Infrastructure as Code come Terraform e Pulumi forniti da AWS Cloud Control API

Ora puoi gestire l'intero stack di Amazon GameLift Servers risorse utilizzando gli strumenti Infrastructure as Code (IaC). Questi strumenti includono AWS CloudFormation e anche strumenti di terze parti come Terraform e Pulumi. Con questo supporto aggiuntivo, ora puoi concentrarti sullo sviluppo del tuo gioco e sfruttare DevOps le strategie per occuparti della gestione delle risorse, della CI/CD e dell'implementazione per i tuoi clienti.

Ora puoi anche effettuare il provisioning e configurare tutti i tipi di Amazon GameLift Servers risorse utilizzando l'API AWS Cloud Control. Puoi continuare a lavorare con le risorse utilizzando Amazon GameLift Servers APIs o i AWS CloudFormation modelli per Amazon GameLift Servers.

Per i dettagli sulle Amazon GameLift Servers risorse disponibili tramite IAc, vedere il riferimento sul [tipo di Amazon GameLift Servers risorsa di riferimento](#) Amazon GameLift Servers sul tipo di risorsa.

Inoltre, ora puoi ridimensionare automaticamente le tue flotte utilizzando i AWS CloudFormation modelli o l'API AWS Cloud Control utilizzando la nuova proprietà [Fleet](#). `ScalingPolicies`

L'API Cloud Control offre agli sviluppatori un set standard di risorse APIs per creare, leggere, aggiornare, eliminare ed elencare le risorse (CRUDL) su centinaia di AWS servizi e diversi strumenti di terze parti come Terraform e Pulumi.

Ulteriori informazioni:

- [AWS CloudFormation](#)
- [AWS API Cloud Control](#)
- [AWS Fornitore CC Terraform](#)
- [Pulumi](#)

16 novembre 2023: Amazon GameLift Servers aggiorna il plugin standalone per Unity

Versioni SDK aggiornate: Amazon GameLift Servers plugin per Unity, versione 2.0.0

Il Amazon GameLift Servers plug-in per Unity fornisce strumenti e flussi di lavoro che semplificano i passaggi per rendere operativo il gioco Unity per l'hosting su cloud. Amazon GameLift Servers Amazon GameLift Servers è un servizio completamente gestito che consente agli sviluppatori di giochi di gestire e scalare server di gioco dedicati per giochi multiplayer basati su sessioni.

Con questa versione, il plug-in per Unity viene aggiornato per utilizzare le Amazon GameLift Servers funzionalità più recenti, tra cui la versione 5.x dell'SDK del server e il supporto per i test locali con Anywhere. Amazon GameLift Servers Il plugin è compatibile con le versioni Unity 2021.3 LTS e 2022.3 LTS.

Le funzionalità principali del plug-in includono:

- Flussi di lavoro guidati dell'interfaccia utente nell'editor Unity per i seguenti scenari:
 - Metti alla prova la tua integrazione di gioco Amazon GameLift Servers usando la tua workstation locale come host. Questo flusso di lavoro ti aiuta a configurare una flotta Amazon GameLift Servers Anywhere per il tuo computer locale, avviare istanze del server e del client di gioco, richiedere una sessione di gioco e partecipare al gioco. Amazon GameLift Servers

- Implementa una soluzione di hosting cloud per il tuo server di gioco integrato con EC2 Amazon GameLift Servers gestito e risorse di supporto. AWS Questo flusso di lavoro ti aiuta a configurare il gioco per l'hosting su cloud e offre tre scenari di implementazione:
 - Distribuisci il server di gioco su una singola flotta.
 - Distribuisci il server di gioco su una serie di flotte Spot a basso costo in più regioni. AWS
 - Distribuisci il server di gioco con un matchmaker. FlexMatch
- Possibilità di configurare profili utente che si collegano a un utente dell' AWS account e impostare una regione predefinita AWS . Puoi gestire più profili per lavorare in AWS account, utenti di account e aree geografiche diversi.
- Comodità speciali che aiutano a semplificare i processi di Amazon GameLift Servers integrazione e implementazione, tra cui:
 - Ogni soluzione di hosting include AWS risorse di supporto, tra cui un pool di utenti Amazon Cognito che fornisce una convalida unica per giocatore IDs e giocatore. Le soluzioni includono anche un bucket Amazon S3 per lo storage, la notifica degli eventi di Amazon SNS, AWS Lambda funzioni e altre risorse.
 - Per il flusso di lavoro Anywhere, il plug-in automatizza le impostazioni dei parametri del server richieste.
 - Per il flusso di lavoro di Amazon EC2, ogni soluzione di implementazione fornisce un servizio di backend client integrato che utilizza le funzioni Lambda. Il servizio di backend si colloca tra il client di gioco e il Amazon GameLift Servers servizio e gestisce tutte le chiamate dirette al servizio. Amazon GameLift Servers
- Contenuti per i test di integrazione, tra cui risorse e codice per un semplice gioco multiplayer di esempio per illustrare l'integrazione tra server di gioco e client di gioco.
- Documentazione del plug-in con linee guida dettagliate sull'integrazione e codice di esempio.

Tutti gli scenari di implementazione, inclusi quelli per le flotte Anywhere e Amazon EC2, utilizzano AWS CloudFormation modelli per descrivere e distribuire AWS le risorse per la soluzione del gioco. Questi modelli sono inclusi nel download del Amazon GameLift Servers plugin. Puoi usarli così come sono o personalizzarli per il tuo gioco.

Ulteriori informazioni:

- [Amazon GameLift Serversplugin per Unity \(server SDK 5.x\)](#), Guida Amazon GameLift Servers per gli sviluppatori
- [Scarica il plugin da GitHub](#)

- [Informazioni sull'Amazon GameLift Servershosting](#)
- [Forum Amazon GameLift Servers](#)

2 novembre 2023: Amazon GameLift Servers aggiunge il supporto per le credenziali condivise

Versioni SDK aggiornate: SDK 1.11.193 AWS

La nuova funzionalità di credenziali Amazon GameLift Servers condivise consente alle applicazioni distribuite su flotte EC2 gestite di interagire con altre risorse. AWS Questo aggiornamento riguarda le applicazioni che raggruppi e distribuisce insieme ai file binari dei server di gioco integrati con la versione 5.x o successiva dell'SDK del server. (Gli eseguibili dei server di gioco possono già richiedere le credenziali utilizzando l'azione SDK 5.x del server). `GetFleetRoleCredentials()`

Ad esempio, se desideri distribuire la build del tuo server di gioco con un CloudWatch agente Amazon per raccogliere i parametri delle istanze EC2 e altri dati, l'agente deve essere autorizzato a interagire con le tue risorse. CloudWatch Per fare ciò, devi prima impostare un ruolo (AWS Identity and Access Management IAM) con le autorizzazioni per utilizzare le CloudWatch risorse, quindi configurare una flotta con il ruolo IAM e le credenziali condivise abilitati. Quando Amazon GameLift Servers distribuisce la build del server di gioco su ciascuna istanza EC2, genera un file di credenziali condiviso e lo archivia sull'istanza. Tutte le applicazioni sull'istanza possono utilizzare le credenziali condivise. Amazon GameLift Serversaggiorna automaticamente le credenziali temporanee per tutta la durata dell'istanza.

Puoi abilitare le credenziali condivise quando crei una flotta EC2 gestita utilizzando i seguenti metodi:

- Nel flusso di lavoro per la creazione del parco Amazon GameLift Servers console.
- Quando si chiama l'operazione API del servizio `CreateFleet` utilizzando il nuovo parametro `InstanceRoleCredentialsProvider`.
- Quando si chiama l'operazione AWS CLI `aws gamelift create-fleet` con il parametro `instance-role-credentials-provider`

Ulteriori informazioni:

- [Comunica con altre AWS risorse del tuo parco veicoli, Guida per gli](#) sviluppatori Amazon GameLift Servers
- [CreateFleet InstanceRoleCredentialsProvider](#), Riferimento Amazon GameLift ServersAPI

- [Configura un ruolo di servizio IAM](#), Guida per Amazon GameLift Servers gli sviluppatori

28 settembre 2023: Amazon GameLift Servers rilascia un nuovo plug-in standalone per Unreal Engine Engine

Versioni SDK aggiornate: Amazon GameLift Servers plugin per la versione 1.0.0 di Unreal Engine

Il Amazon GameLift Servers plug-in per Unreal Engine fornisce strumenti e flussi di lavoro che semplificano i passaggi necessari per avviare e utilizzare un gioco per l'hosting su cloud. Amazon GameLift Servers è un servizio completamente gestito che consente agli sviluppatori di giochi di gestire e scalare server di gioco dedicati per giochi multiplayer basati su sessioni. Il plugin supporta le versioni UE 5.0, 5.1 e 5.2. Le caratteristiche principali includono:

- I flussi di lavoro guidati dell'interfaccia utente nell'editor Unreal] seguono i seguenti percorsi:
 - Metti alla prova l'integrazione del gioco Amazon GameLift Servers usando la tua workstation locale come host. Questo flusso di lavoro ti aiuta a configurare una flotta Amazon GameLift Servers Anywhere per il tuo computer locale, avviare istanze del server e del client di gioco, richiedere una sessione di gioco e ottenere informazioni di connessione per la nuova sessione di gioco. Amazon GameLift Servers
 - Implementa una soluzione di hosting cloud Amazon EC2 per il tuo server di gioco integrato. Questo flusso di lavoro ti aiuta a configurare il gioco per l'hosting su cloud e offre tre diversi scenari di implementazione: distribuzione su una singola flotta, distribuzione su una serie di flotte spot in più regioni o distribuzione su un set di flotte con un matchmaker. FlexMatch La soluzione per ogni scenario di implementazione include risorse e risorse di supporto. Amazon GameLift Servers AWS
- Possibilità di configurare profili utente collegati a un AWS account utente e definire una AWS regione predefinita. È possibile gestire più profili per lavorare in AWS account, utenti di account e aree geografiche diversi.
- Comodità speciali che aiutano a semplificare i processi di Amazon GameLift Servers integrazione e implementazione, tra cui:
 - Ogni soluzione di hosting include AWS risorse di supporto, tra cui un pool di utenti Amazon Cognito di base che fornisce un player unico IDs, un bucket Amazon S3 per lo storage, notifiche di eventi Amazon SNS e funzioni. AWS Lambda
 - Per il flusso di lavoro Anywhere, il plug-in automatizza le impostazioni dei parametri del server richieste utilizzando argomenti della riga di comando.

- Per il flusso di lavoro di Amazon EC2, ogni soluzione di implementazione fornisce un servizio di backend client integrato che utilizza le funzioni Lambda. Il servizio di backend riceve le richieste dai client di gioco e le trasmette al servizio. Amazon GameLift Servers
- Contenuti per i test di integrazione, tra cui una mappa di gioco iniziale e due mappe di test con progetti di base ed elementi dell'interfaccia utente.
- Documentazione del plug-in con linee guida dettagliate sull'integrazione e codice di esempio.

Tutti gli scenari di implementazione, inclusi quelli per le flotte Anywhere e Amazon EC2, utilizzano AWS CloudFormation modelli per descrivere le soluzioni. Il plugin utilizza questi modelli per distribuire Amazon GameLift Servers risorse per il gioco. Questi modelli sono inclusi nel download del Amazon GameLift Servers plugin e sono modificabili. Puoi usarli così come sono o modificarli per il tuo gioco.

Ulteriori informazioni:

- [Amazon GameLift Serversplugin per Unreal Engine](#), Guida Amazon GameLift Servers per gli sviluppatori
- [Scarica il plugin da GitHub](#)
- [Informazioni sull'Amazon GameLift Servershosting](#)
- [Forum Amazon GameLift Servers](#)

17 agosto 2023: Amazon GameLift Servers offre hosting di server di gioco con processori AWS Graviton

Versioni SDK aggiornate: SDK 1.11.144 AWS

Ora Amazon GameLift Servers puoi ospitare i tuoi giochi nel cloud utilizzando istanze EC2 con processori Graviton. AWS Progettate AWS con processori basati su ARM64, le istanze Graviton offrono il miglior rapporto prezzo/prestazioni per i carichi di lavoro cloud che utilizzano EC2, con un miglioramento fino al 40% rispetto alle istanze simili basate su x86. I più recenti processori Graviton3 offrono prestazioni di elaborazione migliori fino al 25% rispetto alle versioni precedenti.

ConAmazon GameLift Servers, ora puoi scegliere tra queste nuove istanze della famiglia Graviton:
AWS

- Istanze basate su Graviton2: c6g, c6gn, r6g, m6g, g5g
- Istanze basate su Graviton3: c7g, r7g, m7g

Ulteriori informazioni:

- [AWS Processore Graviton](#): scopri i vantaggi e gli usi pratici delle istanze EC2 basate su Graviton.
- [Guida introduttiva a Graviton](#): ottieni una panoramica delle istanze basate su Graviton e approfondimenti su come le applicazioni vengono eseguite su di esse a seconda del sistema operativo, delle lingue e dei tempi di esecuzione.

Note

Le istanze Graviton Arm richiedono una build del server per un'AMI Linux. Per C++ e C# è richiesto Server SDK 5.1.1 o versione successiva. Per Go è richiesto Server SDK 5.0 o versione successiva. Queste istanze non forniscono out-of-the-box supporto per l'installazione di Mono su Amazon Linux 2023 (AL2023) o Amazon Linux 2 (AL2).

27 luglio 2023: Amazon GameLift Servers rilascia il server SDK 5.1.0 con supporto aggiuntivo per lo sviluppo di Unity

Versioni SDK aggiornate: Server SDK for C++, C#/Unity, Unreal 5.1.0

La versione più recente del Amazon GameLift Servers server SDK fornisce aggiornamenti per C++, C# e il plug-in Unreal e un nuovo plug-in da utilizzare con il motore di gioco Unity. Gli sviluppatori di giochi integrano l'SDK del Amazon GameLift Servers server nei server di gioco su cui distribuiscono per l'hosting. Amazon GameLift Servers

L'ultima versione dell'SDK del server contiene i seguenti aggiornamenti, che includono una serie di richieste dei clienti:

- Scarica pacchetti SDK specifici per lingua: il [sito di Amazon GameLift Servers download](#) aggiornato contiene pacchetti SDK per ogni lingua. Puoi scaricare le versioni correnti o precedenti.
- Nuovo plug-in SDK per server C# per Unity: il nuovo pacchetto SDK server per Unity contiene librerie C# integrate che puoi installare utilizzando il gestore di pacchetti in Unity Editor (vedi la nuova guida all'integrazione di [Unity](#)). Queste librerie includono le dipendenze richieste tramite UnityNuGet. Puoi utilizzare questo plugin con Unity 2020.3 LTS, 2021.3 LTS e 2022.3 LTS per Windows e Mac OS. Supporta i profili .NET Framework e .NET Standard di Unity, con .NET Standard 2.1 e .NET 4.x.

- Soluzione .NET consolidata per C#: il server SDK per C# ora supporta .NET Framework 4.6.2 (aggiornato dalla versione 4.6.1) e .NET 6.0 in un'unica soluzione. .NET Standard 2.1 è disponibile con le librerie create da Unity.
- Aggiornamenti Server SDK 5.1.0
 - [C++, C#, Unreal] Ora puoi chiamare `InitSDK()` con o senza i parametri del server. I server di gioco che funzionano su flotte EC2 Amazon GameLift Servers gestite leggono i parametri del server direttamente dalle variabili di ambiente. I server di gioco sulle flotte Amazon GameLift Servers Anywhere devono effettuare chiamate `InitSDK()` con i parametri del server.
 - [C++, C#, Unreal] Le chiamate Server SDK hanno migliorato la messaggistica di errore.
 - [C++ SDK] Per migliorare i tempi di compilazione di Server SDK, il flag di compilazione è disabilitato per impostazione predefinita. `-DRUN_CLANG_FORMAT` Puoi abilitarlo con. `-DRUN_CLANG_FORMAT=1`
 - [C++ SDK] Quando si creano le librerie senza le librerie standard (`-DGAMELIFT_USE_STD=0`), `InitSDK()` non utilizza più i tipi di `std::` dati.
- Documentazione estesa del server SDK 5.x
 - Guide di riferimento SDK del server aggiornate per C++, C#/Unity e Unreal, inclusa una copertura estesa di tutti i tipi di dati.
 - [Server C# SDK 5.x per -- Azioni Amazon GameLift Servers](#)
 - [Server C++ SDK 5.x per -- Azioni Amazon GameLift Servers](#)
 - [Server C++ \(Unreal\) SDK 5.x per -- Azioni Amazon GameLift Servers](#)
 - Nuove versioni delle guide all'integrazione dell'SDK 5 del server per i plugin Unity e Unreal
 - [Integrazione Amazon GameLift Servers in un progetto Unity](#)
 - [Integrazione Amazon GameLift Servers in un progetto Unreal Engine](#)
- Aggiornamenti della documentazione aggiuntivi
 - Documentazione rivista per le operazioni delle API di Amazon GameLift Servers servizio [GetComputeAccess](#) [GetInstanceAccess](#) per chiarire le procedure di accesso remoto in base alla versione SDK del Amazon GameLift Servers server in uso.
 - Descrizioni riviste per [GameSessionPlacement](#) documentare come le informazioni sulla sessione di gioco siano transitorie quando un posizionamento è in stato «in sospeso».

13 luglio 2023: Amazon GameLift Servers aggiunge le metriche hardware della flotta

Ora puoi tenere traccia delle metriche delle prestazioni hardware per le tue flotte EC2 Amazon GameLift Servers gestite. Le metriche includono le metriche delle istanze EC2 per l'utilizzo della CPU, il volume del traffico di rete e l'attività del disco. read/write Infatti Amazon GameLift Servers, queste metriche descrivono tutte le istanze attive in una flotta di sedi. Puoi visualizzare queste metriche hardware della flotta utilizzando una CloudWatch dashboard di Amazon nel Console di gestione AWS. Puoi anche visualizzarli nella Amazon GameLift Servers console nei dettagli del parco veicoli.

Ulteriori informazioni:

- [Monitora Amazon GameLift Servers con Amazon CloudWatch](#)(Metriche per le flotte), Amazon GameLift Servers Guida per gli sviluppatori

29 giugno 2023: Amazon GameLift Servers avvia il supporto per Amazon Linux 2023

Versioni SDK aggiornate: SDK 1.11.111 AWS

Amazon GameLift Servers i clienti possono ora utilizzare il sistema operativo Amazon Linux 2023 per ospitare i propri server di gioco. AL2023 offre diversi miglioramenti rispetto all' AL2 inclusione della sicurezza. Questo sistema operativo è disponibile in tutte le Regioni AWS regioni ad eccezione delle regioni della Cina.

I clienti possono utilizzare i più recenti sistemi operativi Linux e continuare a ricevere aggiornamenti di sicurezza critici quando il supporto per Amazon Linux (AL1) terminerà a dicembre 2023. Il supporto per Amazon Linux 2 continuerà fino al 30 giugno 2026.

Ulteriori informazioni:

- [Amazon GameLift Servers Domande frequenti su Linux Server](#)
- [Confronto tra Amazon Linux 2 e Amazon Linux 2023](#)
- Amazon GameLift Servers Link di riferimento API:
 - [AWS Azione SDK CreateBuild](#)
 - [Comando CLI upload-build](#)
 - [Comando CLI create-build](#)

25 maggio 2023: Amazon GameLift Servers FleetIQ aggiunge un filtro per escludere i posizionamenti delle sessioni di gioco sulle istanze che si esauriscono

Versioni SDK aggiornate: SDK 1.11.87 AWS

Se lo utilizzi Amazon GameLift Servers FleetIQ per l'hosting di giochi, ora puoi impedire il posizionamento delle sessioni di gioco su istanze attualmente in fase di esaurimento. Le istanze in fase di esaurimento sono contrassegnate per essere chiuse, ma possono comunque essere selezionate per ospitare nuove sessioni di gioco se non sono disponibili altre risorse di hosting. Con questa nuova funzionalità, puoi escludere completamente l'uso di istanze drenanti.

Usa questa funzione quando chiami `ClaimGameServer` per trovare i server di gioco disponibili. Aggiungi il nuovo `FilterOption` parametro e imposta gli stati delle istanze consentiti solo su `ACTIVE`. In risposta, Amazon GameLift Servers FleetIQ esamina solo le istanze attive durante la ricerca e la rivendicazione di un server di gioco disponibile.

Ulteriori informazioni:

- [ClaimGameServer](#) nel Documentazione di riferimento API di Amazon GameLift Servers
- [Come FleetIQ funziona nella Guida](#) per gli sviluppatori Amazon GameLift Servers FleetIQ

16 maggio 2023: Amazon GameLift Servers supporta l'etichettatura dell'allocazione dei costi per le flotte

Amazon GameLift Servers clienti possono ora utilizzare i tag di allocazione AWS Billing dei costi per organizzare i costi di hosting dei giochi. Puoi assegnare tag di allocazione dei costi alle singole risorse della flotta Amazon GameLift Servers EC2 per tenere traccia del contributo delle flotte ai costi complessivi di hosting.

Ulteriori informazioni:

- [Costi delle risorse e strumenti di utilizzo](#)
- [Utilizzo dei tag di allocazione AWS dei costi](#), Guida per l'utente AWS Billing

20 aprile 2023: Amazon GameLift Servers avvia il supporto per Windows Server 2016

Versioni SDK aggiornate: SDK 1.11.63 AWS

Amazon GameLift Servers i clienti possono ora utilizzare il sistema operativo Windows Server 2016 per ospitare i propri server di gioco. Questo sistema operativo è disponibile in tutte le versioni Regioni AWS. I clienti possono utilizzare il nuovo sistema operativo Windows e continuare a ricevere aggiornamenti di sicurezza critici poiché Microsoft terminerà il supporto per Windows Server 2012 a ottobre 2023.

A partire da oggi, i nuovi clienti che richiedono un ambiente di runtime Windows devono specificare Windows Server 2016 quando creano nuove build di server di gioco per l'hosting. I clienti esistenti possono continuare a creare nuove build e flotte con Windows Server 2012, ma devono completare la migrazione con Windows Server 2016 prima della data di fine del supporto di Microsoft, il 10 ottobre 2023.

Questo aggiornamento include le seguenti modifiche al servizio:

- Quando crei una build di server di gioco utilizzando i comandi Amazon GameLift Servers SDK o CLI, ora devi impostare in modo esplicito il sistema operativo. Non esiste più un valore predefinito. Per distribuire il tuo server di gioco su Windows Server 2016, usa il valore `WINDOWS_2016`.
- Quando crei una build di server di gioco utilizzando la Amazon GameLift Servers console, devi selezionare un sistema operativo tra i valori disponibili. Se sei un cliente esistente con flotte Windows Server 2012 attive, puoi scegliere una delle due opzioni `WINDOWS_2012` oppure `WINDOWS_2016`.

Ulteriori informazioni:

- Amazon GameLift Servers Link di riferimento API:
 - [Comando CLI `upload-build`](#)
 - [Comando CLI `create-build`](#)
 - [AWS Azione SDK `CreateBuild`](#)
- [Amazon GameLift Servers Domande frequenti per Windows 2012](#)

13 aprile 2023: Amazon GameLift Servers lancia il server SDK 5.x per Unreal

Versioni SDK aggiornate: Server SDK 5.0.0 per Unreal

L'ultima versione del plug-in Amazon GameLift Servers leggero per Unreal Engine è ora basata sul server SDK 5.x. Amazon GameLift Servers Per iniziare a integrare il tuo ambiente Unreal Engine, consulta i Amazon GameLift Servers seguenti link.

Ulteriori informazioni:

- [Integrazione Amazon GameLift Servers in un progetto Unreal Engine](#)
- [Aggiungi Amazon GameLift Servers al tuo server di gioco con l'SDK del server](#)
- [Server C++ SDK 5.x per -- Azioni Amazon GameLift Servers](#)

14 marzo 2023: Amazon GameLift Servers lancia una nuova esperienza per console

La nuova Amazon GameLift Servers console include i seguenti miglioramenti:

- Navigazione migliorata: il nuovo pannello di navigazione facilita la navigazione tra Amazon GameLift Servers le risorse.
- Amazon GameLift Servers pagina di destinazione: la nuova pagina di destinazione fornisce collegamenti a documentazione utile, mostra una panoramica di Amazon GameLift Servers alto livello e fornisce supporto tramite collegamenti alla documentazione, domande frequenti e. AWS re:Post
- CloudWatch Metriche Amazon migliorate: le Amazon GameLift Servers metriche sono ora disponibili sia nella Amazon GameLift Servers console che nelle dashboard. CloudWatch Questo aggiornamento include anche nuove metriche per prestazioni, utilizzo e sessioni di gioco.

Ulteriori informazioni:

- [Gestisci le risorse di hosting di giochi con Amazon GameLift Servers](#)
- [Costruire un matchmaker FlexMatch](#)

14 febbraio 2023: Amazon GameLift Servers ora supporta la crittografia lato server per gli argomenti di Amazon SNS

Server Side Encryption ((SSE)) per argomenti SNS crittografa i dati sensibili quando sono inattivi. SSE utilizza le chiavi AWS Key Management Service (AWS KMS) per proteggere il contenuto degli argomenti SNS.

Ulteriori informazioni:

- [Imposta la notifica degli eventi per il posizionamento della sessione di gioco](#)
- [Eventi di abbinamento di FlexMatch](#)

- [Crittografia dei dati inattivi](#)

9 febbraio 2023: l'SDK Amazon GameLift Servers del server supporta .NET 6 con C#10

Versioni SDK aggiornate: Server SDK 5.0.0 per .NET 6. Non sono richiesti aggiornamenti SDK.

Se utilizzi la piattaforma di sviluppo Unity Real-Time, continua a utilizzare il Amazon GameLift Servers server SDK 5.0.0 con .NET 4.6. Unity non supporta .NET 6.

Ulteriori informazioni:

- Scarica l'ultima versione dell'SDK Amazon GameLift Servers del server all'[Amazon GameLift Servers](#)inizio
- [Server C# SDK 5.x per -- Azioni Amazon GameLift Servers](#)

31 gennaio 2023: l'SDK Amazon GameLift Servers del server supporta il linguaggio Go

Versioni SDK aggiornate: Server SDK 5.0.0 per Go

Ulteriori informazioni:

- [Scarica la versione più recente dell'SDK del Amazon GameLift Servers server all'inizio Amazon GameLift Servers](#)
- [Go server SDK per Amazon GameLift Servers -- Azioni](#)

1 dicembre 2022: Amazon GameLift Servers lancia Amazon GameLift Servers Anywhere e Amazon GameLift Servers Server SDK 5.0

Versioni SDK aggiornate: SDK 1.10.21, Server AWS SDK 5.0.0 per C++ e C#

Amazon GameLift ServersAnywhere utilizza le risorse del server di gioco per ospitare server di gioco. Amazon GameLift Servers Puoi utilizzare Amazon GameLift Servers Anywhere per integrare le tue risorse di elaborazione con l'elaborazione Amazon GameLift Servers gestita di EC2 per distribuire i server di gioco su più tipi di elaborazione. Puoi anche usare Amazon GameLift Servers Anywhere per testare in modo iterativo i tuoi server di gioco senza caricare la build per ogni iterazione. Amazon GameLift Servers

Punti salienti:

- Nuovi tipi di flotta e calcolo Amazon GameLift Servers Anywhere
- Amazon GameLift Servers Registrazione delle risorse di calcolo Anywhere
- Ciclo di iterazione dei test migliorato

Amazon GameLift Servers Server SDK 5.0.0 introduce miglioramenti all'SDK del server esistente e un nuovo tipo di risorsa, compute. Server SDK 5.0.0 supporta Amazon GameLift Servers Anywhere e l'uso delle proprie risorse di elaborazione per l'hosting di server di gioco.

Ulteriori informazioni:

- [Server SDK 5.x per Amazon GameLift Servers](#)
- [Posizioni geografiche](#)
- [Scegli le risorse di elaborazione per una flotta gestita](#)
- [Crea un Amazon GameLift Servers Flotta ovunque](#)

25 agosto 2022: Amazon GameLift Servers lancia il supporto per Local Zones

Versioni SDK aggiornate: SDK 1.9.333 AWS

Amazon GameLift Servers è ora disponibile in otto Local Zones negli Stati Uniti d'America, quindi puoi schierare le tue flotte più vicino ai giocatori. Puoi utilizzare tutte le Amazon GameLift Servers funzionalità gestite con Local Zones aggiungendo le Local Zones alle tue flotte.

Le Local Zones estendono AWS risorse e servizi all'edge del cloud, vicino a grandi centri di popolazione, industria e tecnologia dell'informazione (IT). Ciò significa che è possibile distribuire applicazioni che richiedono una latenza di un millisecondo più vicino agli utenti finali o ai data center locali.

Ulteriori informazioni:

- [Amazon GameLift Servers zone locali](#)
- [Posizioni geografiche](#)
- [Crea una EC2 flotta Amazon GameLift Servers gestita](#)

28 giugno 2022: Amazon GameLift Servers lancia una nuova esperienza di console opt-in

La nuova Amazon GameLift Servers console include i seguenti miglioramenti:

- Navigazione migliorata: il nuovo pannello di navigazione facilita la navigazione tra Amazon GameLift Servers le risorse.
- Amazon GameLift Servers pagina di destinazione: la nuova pagina di destinazione fornisce collegamenti a documentazione utile, mostra una panoramica di Amazon GameLift Servers alto livello e fornisce supporto tramite collegamenti alla documentazione, domande frequenti e. AWS re:Post
- CloudWatch Metriche Amazon migliorate: le Amazon GameLift Servers metriche sono ora disponibili sia nella Amazon GameLift Servers console che nelle dashboard. CloudWatch Questo aggiornamento include anche nuove metriche per prestazioni, utilizzo e sessioni di gioco.

Ulteriori informazioni:

- [Gestisci le risorse di hosting di giochi con Amazon GameLift Servers](#)
- [Costruire un matchmaker FlexMatch](#)

15 febbraio 2022: FlexMatch aggiunge una regola composta e ulteriori miglioramenti

FlexMatch gli utenti ora hanno accesso alle seguenti funzionalità:

- Regola composta: è stato aggiunto il supporto per le regole di matchmaking composte per partite di 40 o meno giocatori. Ora puoi usare istruzioni logiche per creare una regola composta per formare una partita. Senza una regola composta nel set di regole, per creare una corrispondenza, tutte le regole del set di regole devono essere vere. Con le regole composte, è possibile scegliere quali regole applicare utilizzando i seguenti operatori logici: `and`, `or`, `not`, `xor`.
- Selezione flessibile dei team: espressioni delle proprietà di matchmaking aggiornate per supportare la selezione di un sottoinsieme di tutti i team disponibili.
- Elenchi di stringhe più lunghi: è stato aumentato il numero massimo di stringhe da 10 a 100 in un elenco di stringhe di valori degli attributi dei giocatori.

Ulteriori informazioni:

- [Amazon GameLift ServersFlexMatchguida per gli sviluppatori:](#)
 - [FlexMatchtipi di regole](#)
 - [FlexMatchespressioni di proprietà](#)
- [AttributeValue: SL](#)

28 ottobre 2021: Amazon GameLift Servers aggiunge il supporto per le flotte multiregionali nella regione Asia Pacifico (Osaka); Amazon GameLift Servers FleetIQ aggiunge il supporto per i processori Graviton2 AWS

[Versioni SDK aggiornate: SDK 1.9.133 AWS](#)

Amazon GameLift Servers è ora disponibile nella regione Asia Pacifico (Osaka-Locale). Gli sviluppatori di giochi possono ora distribuire istanze a Osaka utilizzando una flotta multiregionale. GameLift

Ora puoi utilizzare i server di gioco ospitati da Graviton2, basati sull'architettura del processore basata su ARM, per ottenere maggiori prestazioni a un costo inferiore rispetto alle opzioni di elaborazione equivalenti basate su Intel.

Punti salienti:

- Amazon GameLift Servers è ora disponibile nella regione Asia Pacifico (Osaka-Locale).
- Amazon GameLift ServersFleetIQI gruppi di server di gioco possono ora essere configurati per gestire le famiglie di istanze Graviton2 c6g, m6g e r6g.

Ulteriori informazioni:

- [Amazon GameLift ServersFlotta multiregionale](#)
- [CreateGameServerGroup](#)
- [AWS processore gravitonico](#)

20 settembre 2021: Amazon GameLift Servers rilascia il plugin per Unity

Il Amazon GameLift Servers plugin per Unity versione 1.0.0 contiene librerie e interfaccia utente nativa che facilitano l'accesso alle Amazon GameLift Servers risorse e l'Amazon GameLift Serversintegrazione nel gioco Unity. Puoi utilizzare il Amazon GameLift Servers plug-in per Unity per accedere Amazon GameLift Servers APIs e distribuire CloudFormation modelli per scenari di gioco

comuni. Il plugin include anche un gioco di esempio che funziona con gli scenari di esempio. Puoi usare Amazon GameLift Servers Local per vedere i messaggi trasmessi tra il client di gioco e il server di gioco e scoprire come interagisce un gioco tipico. Amazon GameLift Servers

Il plugin per Unity supporta Unity 2019.4 LTS e 2020.3 LTS.

Punti salienti:

- Crea, esegui e modifica un gioco di esempio con scenari diversi o creane uno tuo.
- Implementa AWS CloudFormation scenari di esempio per scenari di gioco tipici, tra cui solo autenticazione, flotta a regione singola, flotte multiregionali con coda e matchmaker personalizzato, flotte Spot con coda e matchmaker personalizzato e. FlexMatch

Ulteriori informazioni:

- [Amazon GameLift ServersIntegrazione dei giochi con il plug-in per Unity](#)

30 giugno 2021: FlexMatch aggiunge la regola BatchDistance

È possibile utilizzare il tipo di regola BatchDistance per specificare una stringa o un attributo numerico, offrendo una serie di vantaggi a ciascun segmento.

Punti salienti:

- Per partite di grandi dimensioni (più di 40 giocatori), invece di bilanciare equamente i giocatori solo per abilità, ora puoi ottenere lo stesso equilibrio in base all'abilità, alle modalità e alle mappe. Assicurati che tutti i partecipanti alla partita appartengano a un gruppo di abilità, raggruppa più attributi numerici come campionato o stile di gioco e raggruppa in base ad attributi di stringa come mappa o modalità di gioco. Puoi anche creare espansioni nel tempo. Ad esempio, puoi creare un'espansione per consentire a un livello di abilità più ampio di accedere alla partita man mano che il giocatore aspetta.

Per le partite con meno di 40 giocatori, puoi usare una nuova espressione di regole semplificata.

3 giugno 2021: aggiornamenti dell'SDK del client e dell'SDK del server in Amazon GameLift Servers tempo reale

Versioni SDK aggiornate: Realtime Client SDK 1.2.0, Server SDK 3.4.0 per Unreal

Con questo ultimo aggiornamento SDK, ora puoi integrare IL2 CPP nelle tue applicazioni mobili che utilizzano RTS Client SDK e seguire le migliori pratiche con i framework. Ora puoi anche creare il Amazon GameLift Servers Server SDK per Unreal versione 4.26. Questo aggiornamento contiene componenti che si integrano con il server di gioco Windows o Linux, tra cui le versioni C++ e C# di Amazon GameLift Servers Server SDK, Amazon GameLift Servers Local e un plug-in Unreal Engine.

Punti salienti:

- È stato aggiunto il supporto per IL2 CPP nell'RTS Client SDK e per la creazione di librerie native come framework, in modo da poter creare client RTS per i dispositivi mobili più recenti.
- Puoi utilizzarlo per [DescribePlayerSessions\(\)](#) ottenere informazioni per una sessione a giocatore singolo, per tutte le sessioni di gioco o per tutte le sessioni di giocatore associate a un ID giocatore singolo.
- È possibile [GetInstanceCertificate\(\)](#) utilizzarlo per recuperare la posizione del file di un certificato TLS con codifica PEM associato alla flotta e alle relative istanze.
- Supporto Created Server SDK per Unreal versione 4.26.
- L'SDK C# esistente, versione 4.0.2, è stata verificata la compatibilità con Unity 2020.3. Non sono stati richiesti aggiornamenti SDK.

Ulteriori informazioni:

- [Amazon GameLift Servers Guida per gli sviluppatori:](#)
 - [DescribePlayerSessions\(\)](#)
 - [GetInstanceCertificate\(\)](#)

23 marzo 2021: Amazon GameLift Servers aggiunge notifiche al posizionamento delle sessioni di gioco

Versioni SDK aggiornate: AWS [SDK 1.8.168](#)

Ora puoi utilizzare gli eventi per monitorare l'attività di posizionamento delle sessioni di gioco per una coda di sessione di gioco. Crea un argomento Amazon Simple Notification Service (Amazon SNS) per pubblicare notifiche di eventi o configura il monitoraggio degli eventi tramite Events. CloudWatch

Punti salienti:

- Per ogni coda, puoi impostare una stringa di testo personalizzata da includere in tutti i messaggi relativi agli eventi.
- Quando utilizzi un argomento di Amazon SNS, puoi impostare condizioni di accesso aggiuntive che limitano la pubblicazione a code specifiche.

Ulteriori informazioni:

- Amazon GameLift Servers Guida per gli sviluppatori:
 - [Imposta la notifica degli eventi per il posizionamento della sessione di gioco](#) (nuovo)
 - [Eventi di collocamento delle sessioni di gioco](#) (nuovo)
- [Riferimento all'API \(AWS SDK\)](#)
 - Nuovi parametri di coda delle sessioni di gioco `NotificationTarget` e `CustomEventData: GameSessionQueue`, [CreateGameSessionQueueUpdateGameSessionQueue](#)
- [Forum Amazon GameLift Servers](#)

16 marzo 2021: Amazon GameLift Servers aggiunge flotte multiregionali, sei nuove regioni

[Versioni SDK aggiornate: SDK 1.8.163 AWS](#)

Amazon GameLift Servers l'hosting gestito è ora disponibile in 21 regioni. AWS Le nuove regioni sono Città del Capo (af-south-1), Bahrein (me-south-1), Hong Kong (ap-east-1), Milano (), Parigi (eu-south-1) e Stoccolma (eu-west-3). eu-north-1

Con la nuova funzionalità di flotte con Amazon GameLift Servers più sedi, ora puoi configurare un'unica flotta per ospitare i tuoi server di gioco in una o tutte le 20 regioni Amazon GameLift Servers supportate (esclusa la regione di Pechino). Questa funzionalità mira a ridurre in modo significativo il lavoro necessario per configurare e mantenere Amazon GameLift Servers le risorse di hosting a livello globale. È possibile creare flotte con più sedi AWS nelle seguenti regioni: us-east-1 (Virginia settentrionale), us-west-2 (Oregon), eu-central-1 (Francoforte), (eu-west-1 Irlanda), (Sydney), ap-southeast-2 (Tokyo) e ap-northeast-1 (Seoul). ap-northeast-2 In tutte le altre regioni, puoi continuare a configurare flotte con una sola sede, se necessario. Tutte le flotte create prima di questa versione sono flotte con una sola sede. L'utilizzo di flotte con più sedi non influisce sui costi di hosting. Amazon GameLift Servers i prezzi si basano sul tipo, l'ubicazione e il volume delle istanze utilizzate. (Per ulteriori informazioni, consulta la pagina [Amazon GameLift Servers dei prezzi](#)). CloudFormation il supporto per flotte con più sedi sarà presto disponibile.

 Note

Le flotte con più sedi non sono disponibili nelle regioni della Cina. Amazon GameLift Servers le risorse che risiedono nelle regioni della Cina non possono interagire o essere utilizzate dalle risorse di altre Amazon GameLift Servers regioni.

Punti salienti:

- Con una flotta con più sedi, aggiungi esplicitamente un elenco di località remote. Amazon GameLift Servers distribuisce istanze dello stesso tipo e configurazione, inclusa la configurazione di build e runtime, nella regione di origine della flotta e in tutte le sedi aggiunte.
- Regola le impostazioni di capacità e la scalabilità per ogni sede in modo indipendente. Le politiche di scalabilità automatica si applicano a un'intera flotta, ma puoi attivarle o disattivarle in base alla posizione.
- Inizia nuove sessioni di gioco in località specifiche della flotta. Se utilizzi le code delle sessioni di gioco o il matchmaking per effettuare le sessioni di gioco, ora puoi dare la priorità a dove iniziano le nuove sessioni di gioco in base alla località, al costo di hosting e alla latenza del giocatore.
- Ottieni le metriche di hosting nella Amazon GameLift Servers console, aggregate per tutte le sedi di un parco veicoli o suddivise per ciascuna località del parco veicoli.

Ulteriori informazioni:

- [Blog sulla tecnologia dei giochi di Amazon](#)
- [Riferimento alle API \(AWS SDK\)](#)
 - Nuove operazioni di localizzazione della flotta: [CreateFleetLocations](#), [DescribeFleetLocationAttributes](#), [DescribeFleetLocationCapacity](#), [DescribeFleetLocationUtilization](#), [DeleteFleetLocations](#)
 - Operazioni aggiornate della flotta, con nuovo supporto multisede: [CreateFleet](#), [DescribeFleet](#), [UpdateFleetCapacity](#), [EC2InstanceLimits](#), [DescribeInstances](#), [StopFleetActions](#), [StartFleetActions](#)
 - Operazioni di posizionamento delle sessioni di gioco aggiornate, con nuove funzionalità di priorità e filtro: [CreateGameSessionQueue](#), [DescribeGameSessionQueues](#), [UpdateGameSessionQueue](#)

- Operazioni di creazione di sessioni di gioco aggiornate, con nuovo supporto di localizzazione: [CreateGameSession](#), [DescribeGameSessions](#), [DescribeGameSessionDetailsSearchGameSessions](#)
- [Amazon GameLift ServersGuida per gli sviluppatori](#):
 - [Amazon GameLift Serverssedi di assistenza](#)(aggiornato)
 - [Personalizzazioni delle risorse di hosting](#) (nuovo)
 - [Scalabilità della capacità di hosting di giochi con Amazon GameLift Servers](#)(aggiornato)
 - [Personalizza una coda di sessioni di gioco](#) (nuovo)
 - [Dettagli della flotta nella Amazon GameLift Servers console](#)(aggiornato)
- [Forum Amazon GameLift Servers](#)

9 febbraio 2021: Amazon GameLift Servers estende il supporto per le istanze AMD in modalità autonoma FlexMatch

[Versioni SDK aggiornate: SDK 1.8.139 AWS](#)

Questo rilascio include gli aggiornamenti seguenti:

- Amazon GameLift ServersFleetIQI gruppi di server di gioco possono ora essere configurati per gestire le famiglie di istanze AMD C5a, M5a e R5a. I tipi di istanze Amazon EC2 supportati, elencati per GameServerGroup [InstanceDefinition](#), ora includono quanto segue:
 - c5a.large, c5a.xlarge, c5a.2xlarge, c5a.4xlarge, c5a.8xlarge, c5a.12xlarge, c5a.16xlarge, c5a.24xlarge
 - m5a.large, m5a.xlarge, m5a.2xlarge, m5a.4xlarge, m5a.8xlarge, m5a.12xlarge, m5a.16xlarge, m5a.24xlarge
 - r5a.large, r5a.xlarge, r5a.2xlarge, r5a.4xlarge, r5a.8xlarge, r5a.12xlarge, r5a.16xlarge, r5a.24xlarge

Nota: le istanze AMD per non FleetIQ sono attualmente disponibili per l'uso nella AWS regione Cina (Pechino). Vedi [Disponibilità delle funzionalità e differenze di implementazione](#) in Cina.

- Amazon GameLift Serversl'hosting di giochi gestito ora supporta le istanze AMD nella regione Cina (Pechino), gestite da Sinnet. Le nuove famiglie di istanze AMD includono M5a e R5a. I tipi di istanze EC2 supportati, elencati per flotta [InstanceType](#), ora includono quanto segue:

- m5a.large, m5a.xlarge, m5a.2xlarge, m5a.4xlarge, m5a.8xlarge, m5a.12xlarge, m5a.16xlarge, m5a.24xlarge
- r5a.large, r5a.xlarge, r5a.2xlarge, r5a.4xlarge, r5a.8xlarge, r5a.12xlarge, r5a.16xlarge, r5a.24xlarge
- Amazon GameLift Servers FlexMatch ora può essere utilizzato come soluzione di matchmaking autonoma nella regione della Cina (Pechino), gestita da Sinnet. I clienti possono creare un FlexMatch matchmaker nella regione di Pechino e configurare il parametro su STANDALONE. [FlexMatchMode](#) Per ulteriori informazioni in merito FlexMatch, con hosting Amazon GameLift Servers gestito o con una soluzione non di Amazon GameLift Servers hosting, nella Guida per gli [Amazon GameLift Servers FlexMatch sviluppatori](#).
- Quando configuri le notifiche di eventi per Amazon GameLift Servers FlexMatch, ora puoi designare un argomento Amazon SNS FIFO come destinazione della notifica. Per ulteriori informazioni, consulta:
 - [MatchmakingConfiguration NotificationTarget](#), Riferimento API Amazon GameLift Servers
 - [Imposta la notifica FlexMatch degli eventi](#), Guida per Amazon GameLift Servers FlexMatch gli sviluppatori
 - [Presentazione di Amazon SNS FIFO: First-in-first-out Pub/Sub messaggistica](#), notizie e blog AWS

22 dicembre 2020: l'SDK Amazon GameLift Servers del server supporta Unreal Engine 4.25 e Unity 2020

Versioni SDK aggiornate: Amazon GameLift Servers Server SDK 4.0.2, versione 3.3.3 del plugin Unreal

L'ultima versione di Amazon GameLift Servers Server SDK contiene i seguenti componenti:

- Il plugin Unreal aggiornato è stato aggiornato per garantire la compatibilità con Unreal Engine 4.25. L'API non è stata modificata.
- L'SDK C# esistente, versione 4.0.2, è stata verificata la compatibilità con Unity 2020. Non sono stati richiesti aggiornamenti SDK.

Scarica la versione più recente di Amazon GameLift Servers Server SDK all'inizio [Amazon GameLift Servers](#).

24 novembre 2020: Amazon GameLift Servers FlexMatch ora disponibile per i giochi ospitati ovunque

Versioni SDK aggiornate: AWS [SDK 1.8.95](#)

Amazon GameLift Servers FlexMatch è un servizio di matchmaking personalizzabile per giochi multiplayer. Progettato inizialmente per gli utenti di hosting Amazon GameLift Servers gestito, ora FlexMatch può essere integrato in giochi che utilizzano altri sistemi di hosting peer-to-peer, tra cui elaborazione proprietaria locale e tipi primitivi di cloud computing. I giochi che utilizzano Amazon GameLift Servers FleetIQ per l'hosting di giochi su Amazon EC2 ora possono implementare il matchmaking con FlexMatch.

FlexMatch fornisce un robusto algoritmo di matchmaking e un linguaggio di regole che ti offrono un'ampia libertà di personalizzazione del processo di matchmaking in modo che i giocatori vengano abbinati in base alle caratteristiche chiave del giocatore e alla latenza riportata. Inoltre, FlexMatch offre un flusso di lavoro per le richieste di matchmaking che supporta funzionalità come i party tra giocatori, l'accettazione dei giocatori e il riempimento delle partite. Se utilizzi l'FlexMatch hosting Amazon GameLift Servers gestito o Amazon GameLift Servers Realtime, il matchmaker lo utilizza automaticamente Amazon GameLift Servers per trovare risorse di hosting e iniziare una nuova sessione di gioco per le partite appena formate. Quando lo utilizzi FlexMatch come servizio autonomo, il matchmaker invia i risultati delle partite al tuo gioco, che può quindi iniziare una nuova sessione di gioco utilizzando la tua soluzione di hosting.

Le operazioni API per FlexMatch fanno parte dell'API del Amazon GameLift Servers servizio, che è inclusa nell' AWS SDK e in (). AWS Command Line Interface AWS CLI Questa versione include questi aggiornamenti per supportare il matchmaking autonomo:

- La risorsa API MatchmakingConfiguration presenta le seguenti modifiche:
 - Nuova proprietà, FlexMatchMode indica se il matchmaker viene utilizzato con hosting Amazon GameLift Servers gestito o come matchmaking autonomo.
 - La proprietà non GameSessionQueueArns è richiesta quando FlexMatchMode è impostata su standalone.
 - Queste proprietà non vengono utilizzate con il matchmaking autonomo: AdditionalPlayerCount,,,BackfillMode. GameProperties GameSessionData
- La funzione di riempimento automatico non è disponibile con il matchmaking autonomo.

24 novembre 2020: le istanze AMD sono ora disponibili su Amazon GameLift Servers

[Versioni SDK aggiornate: SDK 1.8.95 AWS](#)

L'elenco dei tipi di istanze Amazon EC2 Amazon GameLift Servers attualmente supportati include tre nuove famiglie di istanze: C5a, M5a e R5a. Queste famiglie sono costituite da istanze AMD ottimizzate per il calcolo alimentate da processori AMD EPYC che funzionano a frequenze fino a 3.3. GHz. Le istanze AMD sono compatibili con la tecnologia x86; i giochi attualmente in esecuzione Amazon GameLift Servers possono essere distribuiti su tipi di istanze AMD senza modifiche. Le nuove istanze sono disponibili nelle seguenti AWS regioni: Stati Uniti orientali (Virginia settentrionale e Ohio), Stati Uniti occidentali (Oregon e California settentrionale), Canada centrale (Montreal), Sud America (San Paolo), UE centrale (Francoforte), UE occidentale (Londra e Irlanda), Asia Pacifico meridionale (Mumbai), Asia Pacifico nord-orientale (Seoul e Tokyo) e Asia Pacifico sud-orientale (Singapore e Sydney).

Le nuove istanze AMD includono:

- c5a.large, c5a.xlarge, c5a.2xlarge, c5a.4xlarge, c5a.8xlarge, c5a.12xlarge, c5a.16xlarge, c5a.24xlarge
- m5a.large, m5a.xlarge, m5a.2xlarge, m5a.4xlarge, m5a.8xlarge, m5a.12xlarge, m5a.16xlarge, m5a.24xlarge
- r5a.large, r5a.xlarge, r5a.2xlarge, r5a.4xlarge, r5a.8xlarge, r5a.12xlarge, r5a.16xlarge, r5a.24xlarge

Ulteriori informazioni:

- [Blog sulla tecnologia dei giochi di Amazon](#)
- [Amazon GameLift Serversprezzi delle istanze](#)
- [Istanze Amazon EC2 con processori AMD EPYC](#)
- [Forum Amazon GameLift Servers](#)

11 novembre 2020: aggiornamento della versione all'SDK del server Amazon GameLift Servers

Versioni SDK aggiornate: Amazon GameLift Servers Server SDK 4.0.2

La nuova versione 4.0.2 di Server SDK corregge un problema noto relativo al funzionamento dell'API. `StartMatchBackfill()` Questa operazione restituisce ora una risposta corretta a una richiesta di match backfill.

Il problema non ha influito sul processo di match backfill e non è stata apportata alcuna modifica al funzionamento di questa funzionalità. Il problema potrebbe aver influito sulla messaggistica di registro e sulla gestione degli errori per le richieste di match backfill.

[Scarica la versione più recente di Amazon GameLift Servers Server SDK subito Amazon GameLift Servers dopo.](#)

5 novembre 2020: nuove personalizzazioni degli FlexMatch algoritmi

FlexMatch gli utenti possono ora modificare i seguenti comportamenti predefiniti per il processo di matchmaking. Queste personalizzazioni sono impostate in un set di regole di matchmaking. Non sono state apportate modifiche a. Amazon GameLift Servers SDKs

- Dai priorità ai ticket di backfill: Puoi scegliere di aumentare o diminuire la priorità dei ticket Match Backfill durante la ricerca di corrispondenze accettabili. Dare priorità ai ticket di riempimento è utile quando è abilitata la funzione di riempimento automatico. Utilizzate la proprietà `algorithm.backfillPriority`
- Preordina per ottimizzare la coerenza e l'efficienza delle partite: configura il tuo matchmaker in modo che preordini il pool di biglietti prima di raggruppare i ticket per la valutazione. Ordinando i ticket in base agli attributi chiave dei giocatori, le partite risultanti tendono ad avere giocatori più simili in questi attributi. Puoi anche aumentare l'efficienza del processo di valutazione preordinando gli stessi attributi utilizzati nelle regole delle partite. Utilizzate la proprietà dell'algoritmo `sortByAttributes` con la `strategy` proprietà impostata su «ordinato».
- Modifica il modo in cui vengono attivati i tempi di attesa per l'espansione: scegli se attivare le espansioni in base alla data del ticket più recente (impostazione predefinita) o più vecchio in una partita incompleta. L'attivazione sul ticket più vecchio tende a completare le partite più velocemente, mentre l'attivazione sul ticket più recente porta a una migliore qualità della partita. Usa la proprietà `algorithm.expansionAgeSelection`

17 settembre 2020: Amazon GameLift Servers aggiorna il server SDK

Versioni SDK aggiornate: Amazon GameLift Servers Server SDK 4.0.1

Il nuovo Server SDK contiene i seguenti aggiornamenti:

- API C# versione 4.0.1
 - L'operazione API non [TerminateGameSession\(\)](#) è più supportata. Sostituiscila con una chiamata [ProcessEnding\(\)](#) per terminare sia una sessione di gioco che il processo del server.
 - È stato risolto un problema noto relativo all'operazione [GetInstanceCertificate\(\)](#).
 - L'operazione restituisce [GetTerminationTime\(\)](#) ora un valore del tipo di dati `AwsDateTimeOutcome`.
- API C++ versione 3.4.1
 - L'operazione non [TerminateGameSession\(\)](#) è più supportata. Sostituiscila con una chiamata per [ProcessEnding\(\)](#) terminare sia una sessione di gioco che il processo del server.
- Plugin Unreal Engine versione 3.3.2
 - L'operazione non [TerminateGameSession\(\)](#) è più supportata. Sostituiscila con una chiamata per [ProcessEnding\(\)](#) terminare sia una sessione di gioco che il processo del server.
 - L'operazione di callback `OnUpdateGameSession` viene aggiunta a per [FProcessParametri](#) supportare il match backfill.

[Scarica la versione più recente di Amazon GameLift Servers Server SDK all'inizioAmazon GameLift Servers.](#)

27 agosto 2020: Amazon GameLift Servers FleetIQ per l'hosting di giochi con Amazon EC2 (disponibilità generale)

[Versioni SDK aggiornate: SDK 1.8.36 AWS](#)

La Amazon GameLift Servers FleetIQ soluzione per l'hosting di giochi a basso costo e basato sul cloud su Amazon EC2 è ora disponibile a tutti. Amazon GameLift Servers FleetIQ offre agli sviluppatori la possibilità di ospitare server di gioco direttamente sulle istanze Spot di Amazon EC2 ottimizzando la loro fattibilità per l'hosting di giochi. Gli sviluppatori di giochi possono utilizzarli Amazon GameLift Servers FleetIQ con nuovi giochi o integrare la capacità dei giochi esistenti. Questa soluzione supporta l'uso di contenitori o altri AWS servizi come AWS Shield e Amazon Elastic Container Service (Amazon ECS).

Questa versione di disponibilità generale include i seguenti aggiornamenti della Amazon GameLift Servers FleetIQ soluzione:

- La nuova operazione API `DescribeGameServerInstances` restituisce informazioni, incluso lo stato, su tutte le istanze attive per un gruppo di server di Amazon GameLift Servers FleetIQ gioco.

- Nuova strategia di bilanciamento `ON_DEMAND_ONLY`, configura un gruppo di server di gioco per utilizzare solo istanze On-Demand. Puoi aggiornare la strategia di bilanciamento di un gruppo di server di gioco in qualsiasi momento, rendendo possibile passare dall'utilizzo delle istanze Spot alle istanze On-Demand secondo necessità.
- I seguenti elementi di anteprima sono stati eliminati per motivi di disponibilità generale:
 - Utilizzo di tasti di ordinamento personalizzati per le risorse del server di gioco. I server di gioco possono essere ordinati in base al timestamp di registrazione.
 - Etichettatura delle risorse del server di gioco.

16 aprile 2020: Amazon GameLift Servers aggiorna il server SDK per Unity e Unreal Engine

Versioni SDK aggiornate: Amazon GameLift Servers Server SDK 4.0.0, Local 1.0.5 Amazon GameLift Servers

La versione più recente dell'SDK del server Amazon GameLift Servers contiene i seguenti componenti aggiornati:

- Versione C# SDK 4.0.0 aggiornata per Unity 2019.
- La versione 3.3.1 del plugin Unreal è stata aggiornata per le versioni 4.22, 4.23 e 4.24 di Unreal Engine.
- Amazon GameLift Servers Versione locale 1.0.5 aggiornata per testare le integrazioni che utilizzano la versione 4.0.0 dell'SDK del server C#.

[Scarica la versione più recente di Server SDK subito dopo. Amazon GameLift Servers](#)
[Amazon GameLift Servers](#)

2 aprile 2020: Amazon GameLift Servers FleetIQ disponibile per l'hosting di giochi su EC2 (anteprima pubblica)

[Versioni SDK aggiornate: SDK 1.7.310 AWS](#)

La caratteristica FleetIQ di Amazon GameLift Servers ottimizza l'uso delle istanze Spot a basso costo per l'hosting di giochi. Questa caratteristica è ora estesa per i clienti che desiderano gestire le proprie risorse di hosting direttamente anziché tramite il servizio Amazon GameLift Servers gestito. Questa soluzione supporta l'uso di contenitori o altri AWS servizi come AWS Shield e Amazon Elastic Container Service (Amazon ECS).

Ulteriori informazioni:

[GameTech post sul blog su](#) Amazon GameLift Servers FleetIQ

19 dicembre 2019: migliore AWS gestione delle Amazon GameLift Servers risorse

Versioni SDK aggiornate: AWS [SDK 1.7.249](#)

Ora puoi sfruttare gli strumenti di gestione delle AWS risorse con le risorse. Amazon GameLift Servers In particolare, a tutte le Amazon GameLift Servers risorse chiave (build, script, flotte, code di sessioni di gioco, configurazioni di matchmaking e set di regole di matchmaking) vengono ora assegnati valori Amazon Resource Name (ARN). Un ARN di risorse fornisce un identificatore coerente che è unico in tutte le regioni. AWS Possono essere utilizzati per creare politiche di autorizzazione specifiche per le risorse (AWS Identity and Access Management IAM). Alle risorse viene ora assegnato un ARN e anche l'identificatore di risorsa preesistente, che non è specifico della regione.

Inoltre, ora le risorse Amazon GameLift Servers supportano l'applicazione di tag. Puoi utilizzare i tag per organizzare le risorse, creare politiche di autorizzazione IAM per gestire l'accesso a gruppi di risorse, personalizzare la ripartizione dei costi, ecc. AWS Quando si gestiscono i tag per le risorse Amazon GameLift Servers, utilizzare le operazioni API Amazon GameLift Servers, `TagResource()`, `UntagResource()` e `ListTagsForResource()`.

Ulteriori informazioni:

- [TagResource](#) nel Documentazione di riferimento API di Amazon GameLift Servers
- [Assegnazione di tag alle risorse AWS](#) nella Documentazione di riferimento generale AWS
- [I nomi delle risorse Amazon](#) nella AWS Guida generale

14 novembre 2019: CloudFormation Nuovi modelli, aggiornamenti nella regione Cina (Pechino)

Versioni SDK aggiornate: AWS [SDK 1.7.210](#)

CloudFormation modelli per Amazon GameLift Servers

Amazon GameLift Serversle risorse possono ora essere create e gestite tramite CloudFormation. I modelli di CloudFormation build e fleet esistenti sono stati aggiornati per allinearli alle risorse attuali e ora sono disponibili nuovi modelli per script, code, configurazioni di matchmaking e set di regole

di matchmaking. CloudFormation i modelli semplificano notevolmente il compito di gestire gruppi di AWS risorse correlate, in particolare quando si distribuiscono giochi in più regioni.

Ulteriori informazioni:

- [Amazon GameLift Servers](#)riferimento al tipo di risorsa nella Guida per l'AWS CloudFormation utente
- [Gestisci le risorse di Amazon GameLift Servers hosting utilizzando CloudFormation](#) nella Guida per gli Amazon GameLift Servers sviluppatori

Le traduzioni sono generate tramite traduzione automatica. In caso di conflitto tra il contenuto di una traduzione e la versione originale in Inglese, quest'ultima prevarrà.