



Guia de hospedagem

Amazon GameLift Servers



Amazon GameLift Servers: Guia de hospedagem

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

As marcas comerciais e imagens comerciais da Amazon não podem ser usadas no contexto de nenhum produto ou serviço que não seja da Amazon, nem de qualquer maneira que possa gerar confusão entre os clientes ou que deprecie ou desprestigie a Amazon. Todas as outras marcas comerciais que não são propriedade da Amazon pertencem aos respectivos proprietários, os quais podem ou não ser afiliados, estar conectados ou ser patrocinados pela Amazon.

Table of Contents

O que é o Amazon GameLift Servers?	1
O que você pode fazer com Amazon GameLift Servers	1
Como trabalhar com o Amazon GameLift Servers	2
Soluções do Amazon GameLift Servers	2
Opções de hospedagem	3
FlexMatch para matchmaking	7
FleetIQ para hospedagem autogerenciada da Amazon EC2	8
Realtime servidores com lógica de servidor personalizável	8
Arquitetura da solução de hospedagem gerenciada	9
Componentes do jogo com hospedagem	9
Recursos da solução de hospedagem	12
Como a Amazon GameLift Servers funciona	13
Hospedar servidores de jogos	13
Execução de sessões de jogos	14
Escalar a capacidade de frota	15
Como monitorar o Amazon GameLift Servers	16
Usando outros AWS recursos	17
Como os contêineres funcionam	17
componentes da frota de contêineres	18
Arquiteturas comuns	19
Atributos principais	21
Como os jogadores se conectam aos jogos	28
Locais de serviço	29
AWS Localizações suportadas	29
Locais de hospedagem gerenciada	34
Localizações para Amazon GameLift Servers qualquer lugar	35
Localizações para Amazon GameLift Servers FlexMatch	36
Amazon GameLift Servers na China	36
Preços	37
Gerar estimativas de preços	38
Gerenciar os custos de hospedagem de jogos	41
Conceitos básicos	44
Antes de começar	44
Opções de integração rápida	44

Opções de desenvolvimento personalizadas	45
Exemplos do Amazon GameLift Servers	45
Exemplo de servidor de jogo personalizado	46
Amazon GameLift ServersRealtimeexemplo	46
Configurar um Conta da AWS	47
Inscreva-se para um Conta da AWS	47
Criar um usuário com acesso administrativo	48
Definir permissões de usuário para Amazon GameLift Servers	49
Configurar o acesso programático, consulte para usuários	50
Configurar o acesso programático para o jogo	52
Exemplo de permissões do IAM	52
Configurar um perfil de serviço do IAM	57
Obter ferramentas de desenvolvimento	61
Para servidores de jogos	61
Para serviços de clientes de jogos	64
Para gerenciamento de recursos	65
Para Amazon GameLift ServersRealtime	65
Tutorial: Integração rápida com o Amazon GameLift Servers invólucro	66
Pré-requisitos	67
Visão geral	67
Etapa 1: Obtenha e construa o invólucro do servidor de jogos	68
Etapa 2: Prepare a versão do seu servidor de jogo	68
Etapa 3: configurar a embalagem para sua frota	69
Etapa 4: faça o upload da versão do servidor do jogo	71
Etapa 5: criar a EC2 frota gerenciada	71
Etapa 6: criar e conectar-se a uma sessão de jogo	73
Etapa 7: Gerencie e monitore sua frota	74
Etapa 8: escale seus servidores de jogos	74
Solução de problemas comuns	76
Próximas etapas	77
Explorar com um plug-in de mecanismo de jogo	78
Fluxo do plug-in	79
Plug-in para Unreal Engine	79
Plug-in para Unity (SDK 5.x de servidor)	113
Plug-in para Unity (SDK 4.x de servidor)	135
Roteiro EC2 de desenvolvimento gerenciado	163

Roteiro de desenvolvimento de contêineres gerenciados	169
Roteiro de desenvolvimento em qualquer lugar	177
Roteiro de desenvolvimento de hospedagem híbrida	183
Preparando jogos para Amazon GameLift Servers	192
Integre jogos com servidores de jogos personalizados	192
Amazon GameLift Servers interações	193
Integrar um servidor de jogos	198
Integrar um cliente de jogo	210
Motores de jogo e Amazon GameLift Servers	216
Projete seu serviço de back-end	241
Autenticar os jogadores	241
Back-end de tecnologia sem servidor	242
WebSocketback-end baseado	244
Configuração para desenvolvimento iterativo	246
Criar um ambiente de teste baseado em nuvem	248
Configurar testes no local	252
Trabalhar com o Agent	258
Configurar testes locais (legado)	260
Configurar Amazon GameLift Servers local	261
Testar um servidor de jogos	261
Testar um servidor e um cliente de jogo	265
Variações com o local	268
Adicionando FlexMatch matchmaking	268
Obter dados de frotas	269
Integrando jogos com Amazon GameLift Servers Realtime	270
O que são Realtime servidores?	270
Gerenciar sessões de jogo	270
Interação entre servidores e clientes	271
Personalizar um servidor	272
Implantação e atualização	272
Integrar um cliente do jogo	273
Personalizando um script Realtime	279
Implantação de software de servidor de jogos	286
Implantar uma compilação de servidor personalizada	286
Implantar para hospedagem gerenciada	287
Implementar para hospedagem Anywhere	287

Criar um pacote dos arquivos de compilação do jogo	287
Adicionar um script de instalação da compilação	288
Criar uma compilação para hospedagem gerenciada	291
Atualizar uma compilação de servidor de jogos para hospedagem gerenciada	296
Criar uma imagem de contêiner	297
Crie uma imagem de contêiner de servidor de jogos	298
Envie uma imagem de contêiner para o Amazon ECR	301
Implantar um Amazon GameLift Servers Realtime script	302
Empacotar os arquivos de script	302
Carregar arquivos de script em um diretório local	302
Carregue arquivos de script no Amazon S3	304
Atualizar arquivos de script	306
Configuração de uma frota de hospedagem	307
Características da frota	307
Como a criação de frotas funciona	308
EC2 Frotas gerenciadas	308
Fluxo de trabalho de criação de EC2 frotas	309
Crie uma EC2 frota gerenciada	310
Personalize suas frotas EC2 gerenciadas	320
Atualizar uma configuração de frota	326
Depurar problemas de frota	330
Conectar-se remotamente às instâncias da frota	334
emparelhamento da VPC	343
Frotas de contêineres gerenciadas	350
Crie uma frota de contêineres	352
Atualize uma frota de contêineres	356
Personalize uma frota de contêineres	358
Criar uma definição de grupo de contêineres	366
Atualizar uma definição de grupo de contêineres	372
Excluir uma definição de grupo de contêineres	375
Frotas Anywhere	376
Fluxo de trabalho de criação da frota Anywhere	378
Criar uma frota Anywhere	378
Resumir uma frota com um alias	388
Criar um alias da	389
Editar um alias	390

Gerenciar o posicionamento de sessões de jogo com filas	392
Características da fila	392
Criar uma fila	393
Personalizar uma fila	397
Práticas recomendadas	398
Definir o escopo de uma fila	399
Criar uma fila de vários locais	400
Priorize o posicionamento de sessões de jogo	402
Avaliar métricas de fila	409
Criar para instâncias spot	410
Configurar notificações de eventos	414
Configurar um tópico do SNS	414
Configurar um tópico do SNS com a criptografia do lado do servidor	416
Configurar EventBridge	417
Tutorial: criar uma fila com instâncias spot	418
Etapa 1: Definir o escopo da fila	418
Etapa 2: Criar uma infraestrutura de frota spot	419
Etapa 3: Atribuir aliases para cada frota	420
Etapa 4: Criar uma fila com destinos	421
Etapa 5: Adicionar limites de latência à fila	423
Resumo	425
Escalar a capacidade de hospedagem	427
Para gerenciar a capacidade da frota no console	428
Definir limites de capacidade de hospedagem	428
Como definir limites de capacidade	429
Definir manualmente capacidade de frota	430
Suspender o ajuste de escala automático	430
Como definir manualmente capacidade de frota	431
Capacidade de frota de ajuste de escala automático	432
Ajuste de escala automático baseado em destino	433
Ajuste de escala automático baseado em regras	435
Escale frotas de contêineres	439
Preparar-se para o lançamento do jogo	442
Preparação	442
Preparar para testes	443
Preparar para o início	443

Planejar pós-lançamento	444
Gerenciando recursos de hospedagem com AWS CloudFormation	445
Práticas recomendadas	445
Usando AWS CloudFormation pilhas	446
Pilhas para um único local	446
Pilhas para várias regiões	448
Atualizar compilações	451
Implantar atualizações de compilação automaticamente	452
Implantar atualizações de compilação manualmente	452
Como funcionam as reversões	453
Monitoramento Amazon GameLift Servers	454
Rastreie a hospedagem de jogos no console	455
Painel de hospedagem	456
Compilações de servidores de jogos	457
Amazon GameLift ServersscriptsRealtime	459
Frotas	460
Detalhes da frota	461
Sessões de jogo e sessões de jogadores	464
Aliases	470
Filas de sessões de jogos	471
Monitor com CloudWatch	474
Dimensões de métricas	474
Métricas de frota	475
Métricas de fila	488
FlexMatch métricas	491
FleetIQ métricas	495
Registrar em log chamadas de API	498
Amazon GameLift Servers informações em CloudTrail	498
Noções básicas Amazon GameLift Servers entradas do arquivo de log	499
Registrando mensagens ao servidor	501
Registro em log para servidores personalizados	502
Registro para Amazon GameLift Servers Realtime	504
Segurança	510
Proteção de dados	511
Criptografia inativa	513
Criptografia em trânsito	513

Privacidade do tráfego entre redes	513
Gerenciamento de identidade e acesso	514
Público	514
Autenticar com identidades	515
Gerenciar o acesso usando políticas	519
Como Amazon GameLift Servers funciona com o IAM	521
Exemplos de políticas baseadas em identidade	529
Solução de problemas	535
AWS políticas gerenciadas	537
Registro e monitoramento com Amazon GameLift Servers	539
Validação de conformidade	540
Resiliência	541
Segurança da infraestrutura	542
Análise de configuração e vulnerabilidade	543
Práticas recomendadas de segurança	544
Não abra portas para a internet	544
Saiba mais	545
Guias de referência	546
API de serviço (AWS SDK)	546
Gerenciar recursos Amazon GameLift Servers de hospedagem	547
Inicie sessões de jogo e junte-se aos jogadores	551
SDK 5.x do servidor	552
Atualizações do SDK 5.x do servidor	553
Migrar para o SDK 5.x do servidor	554
C++ server SDK	556
SDK do servidor C#	607
SDK do servidor Go	651
SDK do servidor C++ (Unreal): ações	680
SDK 4 e versões anteriores do servidor	726
SDK do servidor para C++: ações	727
SDK do servidor para C#: ações	751
SDK do servidor para Unreal: ações	773
Amazon GameLift ServersreferênciaRealtime	789
Referência de API (C#) do cliente em tempo real	790
Realtimereferência de script	804
Eventos de posicionamento de sessões de jogo	812

Sintaxe do evento de posicionamento	813
PlacementFulfilled	814
PlacementCancelled	816
PlacementTimedOut	817
PlacementFailed	818
Versões da AMI	819
Cotas e endpoints de serviço	820
Sinalizadores de ping UDP	821
Como funcionam os sinalizadores de ping UDP	821
Casos de uso comuns para beacons de ping UDP	821
Obtendo endpoints de beacon	822
Implementando medições de latência	823
Exemplos de código	824
Notas de lançamento e versões do SDK	832
Versões do SDK	832
Notas da versão	841
.....	dccclxxxvii

O que é o Amazon GameLift Servers?

Use Amazon GameLift Servers para implantar, operar e escalar servidores dedicados e de baixo custo na nuvem para jogos multijogador baseados em sessões. Construído com base na infraestrutura de computação AWS global, Amazon GameLift Servers ajuda a fornecer servidores de jogos de alto desempenho e alta confiabilidade, ao mesmo tempo em que dimensiona dinamicamente o uso de recursos para atender à demanda mundial dos jogadores.

O que você pode fazer com Amazon GameLift Servers

Amazon GameLift Servers oferece suporte a esses casos de uso e muito mais:

- Hospede seus próprios servidores de jogos multijogador personalizados na nuvem com EC2 hospedagem Amazon GameLift Servers gerenciada.
- Execute recursos de hospedagem gerenciada de baixo custo usando [instâncias spot do Amazon Elastic Compute Cloud EC2 \(Amazon\)](#).
- Hospede seus servidores de jogos em contêineres para obter flexibilidade em todas as plataformas e oferecer suporte a migrações com Amazon GameLift Servers contêineres gerenciados.
- Crie uma solução de hospedagem híbrida para oferecer suporte à hospedagem and/or local em várias nuvens e, ao mesmo tempo, gerencie as sessões de jogos em um só lugar com Amazon GameLift Servers o Anywhere.
- Crie um sistema robusto de matchmaking para seus jogos multijogador com Amazon GameLift Servers FlexMatch.
- Dimensione sua capacidade de hospedagem gerenciada automaticamente para atender às suas necessidades de jogo com base no uso real do jogador.
- Gerencie seus recursos EC2 computacionais da Amazon para jogos em um só lugar usando Amazon GameLift Servers FleetIQ.
- Crie um ambiente de teste iterativo para suas compilações de servidor e cliente de jogos com Amazon GameLift Servers Anywhere e. EC2
- Para jogos que não precisam de um servidor de jogos personalizado, configure uma solução de servidor leve com. Amazon GameLift Servers Realtime



Tip

[Comece agora com Amazon GameLift Servers](#)

Como trabalhar com o Amazon GameLift Servers

Use essas ferramentas para trabalhar com o Amazon GameLift Servers.

AWS CLI

Use o AWS Command Line Interface (AWS CLI) para fazer chamadas para o AWS SDK, incluindo a API de serviço para Amazon GameLift Servers. Consulte [Introdução ao AWS CLI](#) no Guia do AWS Command Line Interface usuário.

Console do Amazon GameLift Servers

Use o [AWS Management Console for Amazon GameLift Servers](#) para configurar recursos, gerenciar suas implantações de servidores de jogos e monitorar métricas de desempenho e uso. O Amazon GameLift Servers console é uma alternativa de GUI para gerenciar recursos de forma programática ou com o AWS CLI.

Amazon GameLift Servers SDKs

Eles Amazon GameLift Servers SDKs contêm as bibliotecas necessárias para estabelecer a comunicação entre seus clientes de jogos, servidores de jogos e serviços de jogos e o Amazon GameLift Servers serviço. Para obter mais informações, consulte [Obtenha ferramentas Amazon GameLift Servers de desenvolvimento](#).

SDK de cliente para Amazon GameLift Servers Realtime

O SDK do cliente Amazon GameLift Servers Realtime permite que você conecte seu cliente de jogo a um Realtime servidor fornecido por Amazon GameLift Servers, participe de sessões de jogo e permaneça sincronizado com outros jogadores. Baixe o [SDK](#) e saiba mais sobre como fazer chamadas de API com a [API Amazon GameLift Servers Realtime cliente \(C#\)](#).

Soluções do Amazon GameLift Servers

Amazon GameLift Servers oferece uma variedade de soluções para desenvolvedores que estão criando jogos multijogador baseados em sessões.

Soluções para desenvolvedores de jogos

- [Amazon GameLift Servers](#) opções de hospedagem
- [Amazon GameLift Servers FlexMatch](#) para matchmaking
- [Amazon GameLift Servers FleetIQ](#) para hospedagem autogerenciada da Amazon EC2
- [Amazon GameLift Servers Realtime](#) com lógica de servidor personalizável

Amazon GameLift Servers opções de hospedagem

Ao trabalhar com Amazon GameLift Servers para operar seus servidores de jogos, você tem várias opções de onde e como seus servidores de jogos são hospedados. Se você quiser usar os recursos de hospedagem que já possui ou configurar uma hospedagem baseada em nuvem gerenciada por Amazon GameLift Servers, você pode criar uma experiência de hospedagem perfeita para seus jogadores.

[Gerenciado EC2](#)

[Contentores gerenciados](#)

[Hospedagem híbrida](#)

[Hospedagem Anywhere](#)

Gerenciado EC2

Com a EC2 hospedagem Amazon GameLift Servers gerenciada, você pode liberar a maior parte do trabalho de gerenciar seus servidores de jogos. Escolha recursos computacionais de uma ampla seleção de tipos de EC2 instância da Amazon. Integre seus projetos de jogos e deixe Amazon GameLift Servers cuidar dos detalhes. Para saber mais sobre hospedagem gerenciada, consulte [Como funciona o Amazon GameLift Servers](#).

[Comece a desenvolver uma solução de hospedagem Amazon GameLift Servers gerenciada para seu jogo.](#)

Atributos principais

- Hospede jogos multijogador que sejam executados nos sistemas operacionais Amazon Linux ou Windows Server.
- Ofereça experiências de jogo de baixa latência aos jogadores, onde quer que estejam. Implante servidores de jogos globalmente em qualquer uma das Regiões da AWS Zonas Locais Amazon

GameLift Servers compatíveis. Para obter uma lista completa, consulte [Amazon GameLift Servers locais de serviço](#).

- Use o posicionamento Amazon GameLift Servers inteligente das sessões de jogo para que os jogadores sempre tenham a melhor experiência possível como jogador hospedado. Você pode confiar na Amazon GameLift Servers tomada de decisões ou personalizar de acordo com critérios de posicionamento, como custo, latência do jogador e localizações geográficas.
- Escolha como escalar os recursos de hospedagem para atender à demanda dos jogadores. Gerencie a capacidade manualmente ou configure a escalabilidade automática. Com o escalonamento automático baseado em metas, você pode manter um buffer de tamanho dinâmico de capacidade ociosa, o que ajuda a controlar os custos e, ao mesmo tempo, garantir que novos jogadores possam entrar nos jogos com o mínimo de espera.
- Deixe Amazon GameLift Servers implantar e gerenciar seus servidores de jogos baseados em nuvem. Amazon GameLift Servers cria recursos conforme necessário, instala o software do servidor de jogos e inicia automaticamente os processos para hospedar sessões de jogo para os jogadores. Configure o rastreamento de saúde personalizado e permita Amazon GameLift Servers detectar e resolver recursos de baixo desempenho.
- Aproveite os recursos de Amazon GameLift Servers monitoramento para avaliar o desempenho e o uso. Você pode monitorar métricas sobre fatores como desempenho do hardware, eficiência no posicionamento de sessões de jogo e ciclos de vida dos processos do servidor. Você pode acompanhar as sessões ativas do jogo e as sessões dos jogadores para observar o uso ao longo do tempo. Também pode baixar e armazenar os logs de sessões de jogos.
- Para hospedagem de produção, automatize o gerenciamento e as implantações de recursos de hospedagem de jogos usando AWS CloudFormation modelos para Amazon GameLift Servers e o AWS Cloud Development Kit (AWS CDK) Aproveite as ferramentas e serviços de integração e entrega contínuas (CI/CD), como o AWS CodePipeline.

Contentores gerenciados

Amazon GameLift Servers fornece uma solução completa de hospedagem em nuvem para servidores de jogos em contêineres. Com contêineres Amazon GameLift Servers gerenciados, você pode aproveitar os principais benefícios do uso de contêineres, como portabilidade, agilidade e tolerância a falhas. Os recursos a seguir estão disponíveis com frotas de contêineres gerenciadas.

[Comece a desenvolver uma solução de hospedagem Amazon GameLift Servers gerenciada para seu servidor de jogos em contêineres.](#)

Atributos principais

- Desenvolva uma arquitetura personalizada com contêineres leves para executar seu software de servidor de jogos em recursos de hospedagem Amazon GameLift Servers baseados em Linux.
- Use as ferramentas do Docker para criar uma imagem de contêiner baseada em Linux. Armazene imagens para implantação em um repositório do Amazon Elastic Container Registry (Amazon ECR).
- Ofereça experiências de baixa latência aos jogadores implantando recursos da frota de contêineres em qualquer zona local que Região da AWS ofereça suporte. Amazon GameLift Servers Consulte [Amazon GameLift Servers locais de serviço](#).
- Gerencie o ciclo de vida da frota com ferramentas para modelar as versões do servidor de jogos e implantar atualizações da frota.
- Use os recursos de posicionamento da sessão de Amazon GameLift Servers jogo, incluindo filas e FlexMatch matchmaking, para encontrar as melhores partidas possíveis para seus jogadores.
- Teste sua arquitetura de servidor e contêiner de jogos com o Amazon GameLift Servers serviço usando uma frota Anywhere. Teste seu jogo localmente ou em um ambiente de teste baseado em nuvem.
- Acompanhe o desempenho da hospedagem de jogos com métricas de desempenho específicas do contêiner. Monitore a integridade dos recursos da frota usando métricas de hardware.
- Gerencie os recursos da frota de contêineres usando AWS CloudFormation modelos para Amazon GameLift Servers.

Hospedagem híbrida

Use o Amazon GameLift Servers serviço com uma combinação de hospedagem Amazon GameLift Servers gerenciada e hospedagem autogerenciada Anywhere. Uma abordagem híbrida permite que você compile a solução necessária no momento, enquanto também se prepara para onde precisa estar no futuro. Os cenários comuns em que uma solução híbrida faz sentido incluem:

- Expandir a solução de hospedagem para a Nuvem AWS. Complemente os recursos de sua solução de hospedagem existente (hardware local ou outra hospedagem baseada em nuvem) adicionando Amazon GameLift Servers hospedagem gerenciada. Com a hospedagem gerenciada, você pode aumentar a capacidade de hospedagem ou adicionar capacidade de expansão para aumentar a escala verticalmente de forma rápida e pagar somente pelos recursos quando precisar deles. Você também pode aproveitar a presença global do Amazon GameLift Servers serviço para

alcançar mais jogadores em todo o mundo e oferecer a experiência multijogador de baixa latência que eles esperam.

- Preparar-se para a migração para hospedagem baseada na nuvem. Se você está pensando ou planejando migrar para o Nuvem AWS (em vez de atualizar seu próprio hardware), uma solução de hospedagem híbrida é uma maneira viável de fazer a transição tão gradualmente quanto necessário.
- Aumente a latência para jogadores em locais além daqueles atendidos pela Amazon GameLift Servers. Se você já usa hospedagem Amazon GameLift Servers gerenciada, talvez precise oferecer suporte aos jogadores em determinadas situações. Por exemplo, talvez você queira alcançar jogadores em locais inusitadamente remotos ou reduzir significativamente a latência nessas áreas. Adicione locais de hospedagem personalizados e use Amazon GameLift Servers Anywhere para gerenciar esses locais junto com seus recursos de hospedagem gerenciada.

[Comece a desenvolver uma solução de hospedagem Amazon GameLift Servers híbrida para seu jogo.](#)

Atributos principais

- Use os mesmos componentes de cliente e servidor do jogo com recursos de hospedagem gerenciados e autogerenciados. Ofereça uma experiência de jogador unificada em todos os recursos de hospedagem.
- Use os mesmos FlexMatch matchmakers para colocar partidas em todos os recursos de hospedagem.
- Gerencie centralmente os recursos de hospedagem híbrida enquanto os implanta globalmente.
- À medida que a demanda dos jogadores flutua, gerencie as cargas das sessões de jogo sem problemas nos recursos gerenciados e autogerenciados.
- Com o Amazon GameLift Servers Agente, você pode usar as mesmas ferramentas para gerenciar os ciclos de vida do servidor de jogos em todos os tipos de recursos de hospedagem.
- Reúna métricas e logs de jogos e jogadores em todos os recursos de hospedagem. Aproveite os Amazon GameLift Servers recursos e outros AWS serviços para combinar dados e desenvolver soluções de observabilidade coesas.

Hospedagem Anywhere

Use as frotas do Amazon GameLift Servers Anywhere com gerenciamento de sessões de Amazon GameLift Servers jogo, incluindo matchmaking, para hospedar seus servidores de jogos

personalizados onde você quiser. Anywhereas frotas são particularmente úteis como ambientes de teste para o desenvolvimento rápido e iterativo de jogos. Configure uma frota Anywhere para sua própria estação de trabalho local ou um conjunto de recursos de hospedagem baseados na nuvem. Para hospedagem de produção, você pode usar uma abordagem híbrida com Anywhere frotas para seu hardware local complementadas por Amazon GameLift Servers frotas gerenciadas.

Para obter mais informações sobre testes com o Anywhere, consulte [Configure testes locais com o Amazon GameLift Servers Anywhere](#). Para obter mais informações sobre como configurar uma frota Anywhere, consulte [Configurando uma frota de hospedagem com Amazon GameLift Servers](#).

[Comece a desenvolver uma solução de hospedagem Amazon GameLift Servers em qualquer lugar para seu jogo.](#)

Atributos principais

- Execute testes rápidos e iterativos ao desenvolver os jogos multijogador.
- Use Amazon GameLift Servers ferramentas para gerenciar servidores de jogos hospedados em seu próprio hardware.
- Aproveite o hardware disponível mais próximo dos jogadores, em qualquer lugar.

Amazon GameLift ServersFlexMatchpara matchmaking

Use FlexMatch para criar conjuntos de regras personalizados para definir partidas multijogador para seu jogo. FlexMatchusa conjuntos de regras para comparar jogadores compatíveis em cada partida e fornecer aos jogadores a experiência multijogador ideal.

Para obter mais informações sobreFlexMatch, consulte [O que é Amazon GameLift ServersFlexMatch?](#)

Atributos principais

- Equilibre a velocidade de criação da partida e a qualidade da partida.
- Combine jogadores ou equipes com base em características definidas.
- Defina regras para colocar os jogadores nas partidas com base na latência.

Amazon GameLift ServersFleetIQpara hospedagem autogerenciada da Amazon EC2

Use grupos de servidores de FleetIQ jogos para trabalhar diretamente com seus recursos de hospedagem na Amazon EC2 e no Amazon EC2 Auto Scaling. Isso oferece o benefício de Amazon GameLift Servers otimizações para hospedagem de jogos econômica e resiliente. Essa solução é para desenvolvedores de jogos que precisam de mais flexibilidade do que as Amazon GameLift Servers soluções totalmente gerenciadas oferecem.

Para obter informações sobre como FleetIQ funciona com a Amazon EC2 e o EC2 Auto Scaling para hospedagem de jogos, consulte o Guia do [Amazon GameLift ServersFleetIQdesenvolvedor](#).

Atributos principais

- Obtenha o balanceamento otimizado de instâncias spot usando o FleetIQ algoritmo.
- Use os recursos de roteamento de jogadores para gerenciar os recursos do servidor de jogos com eficiência e fornecer uma melhor experiência ao jogador ao participar de jogos.
- Escale automaticamente a capacidade de hospedagem com base no uso do jogador.
- Gerencie diretamente as EC2 instâncias da Amazon por conta própria Conta da AWS.
- Use qualquer um dos formatos executáveis de servidor de jogo compatíveis, incluindo Windows, Linux, contêineres e Kubernetes.

Amazon GameLift ServersRealtimecom lógica de servidor personalizável

Use Realtime servidores para hospedar jogos que não precisam de um servidor de jogos personalizado. Esta solução de servidor leve fornece servidores de jogos que você poderá configurar para se adequar ao jogo. Você pode hospedar Realtime servidores usando uma solução de hospedagem Amazon GameLift Servers gerenciada.

Para obter mais informações sobre hospedagem com Amazon GameLift ServersRealtime, consulte [Integrando jogos com Amazon GameLift ServersRealtime](#).

Atributos principais

- Use recursos Amazon GameLift Servers de gerenciamento, incluindo escalonamento automático, filas em vários locais e posicionamento de sessões de jogos.
- Use recursos de Amazon GameLift Servers hospedagem e escolha o tipo de hardware de AWS computação para suas frotas.

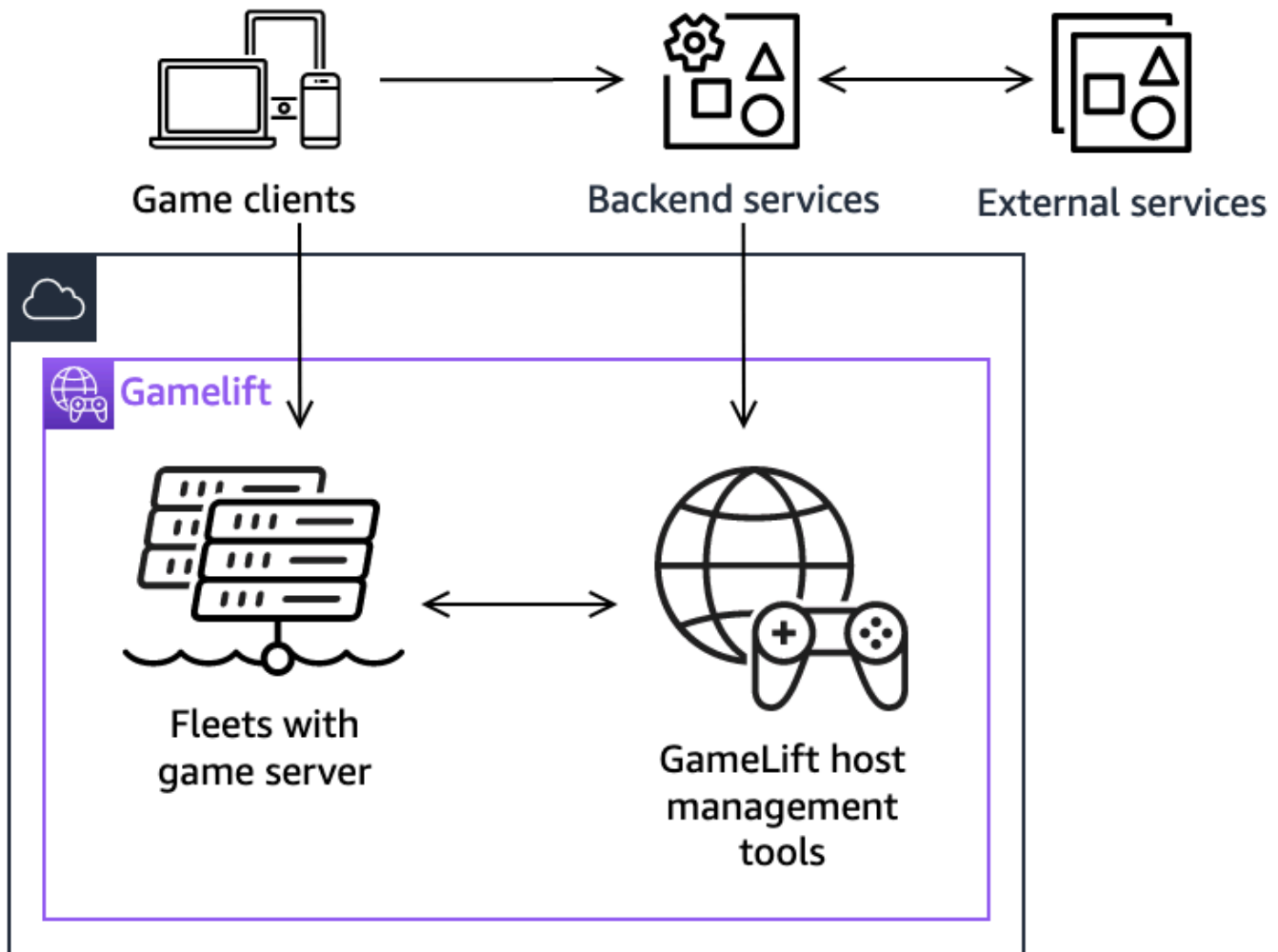
- Aproveite uma pilha de rede completa para interação com o cliente e servidor de jogos.
- Obtenha a funcionalidade principal do servidor de jogos com lógica de servidor personalizável.
- Faça atualizações em tempo real das configurações do Realtime e da lógica de servidor.

Arquitetura de Amazon GameLift Servers solução gerenciada

Os diagramas neste tópico descrevem como uma solução de hospedagem completa Amazon GameLift Servers é estruturada.

Componentes do jogo com hospedagem

O diagrama a seguir ilustra como os principais componentes de uma solução de Amazon GameLift Servers hospedagem gerenciada trabalham juntos para executar servidores de jogos dedicados e ajudar os jogadores a encontrar e se conectar às sessões de jogos hospedadas. A solução de hospedagem que você desenvolve para o jogo incluirá a maioria ou todos esses componentes.



Os principais componentes desta arquitetura incluem o seguinte:

Cientes do jogo

Um cliente do jogo é o software que está sendo executado no dispositivo de um jogador. O jogador joga o jogo entrando em uma sessão de jogo em um servidor de jogos hospedado. Um cliente de jogo pede para entrar em uma sessão por meio de um serviço de backend, recebe informações de conexão de uma sessão de jogo e as usa para se conectar diretamente à sessão. Para obter mais informações, consulte [Preparando jogos para Amazon GameLift Servers](#). Ao se conectar a um Realtime servidor, um cliente de jogo A usa o SDK do cliente para Amazon GameLift ServersRealtime.

Serviços de backend

Um serviço de back-end é um serviço personalizado que você cria para lidar com a comunicação com o Amazon GameLift Servers serviço em nome de um cliente de jogo. Também é possível usar serviços de backend para tarefas específicas do jogo, como autenticação e autorização do jogador, inventário ou controle de moeda. Um serviço de back-end se comunica com o Amazon GameLift Servers serviço usando as operações de API no AWS SDK.

Um serviço de backend faz solicitações para obter informações da sessão de jogo existente e iniciar sessões de jogo. As solicitações de novas sessões de jogo definem certas características, como o número máximo de jogadores. Essas solicitações solicitam Amazon GameLift Servers o início do processo de colocação da sessão do jogo. Quando uma sessão de jogo estiver pronta para aceitar jogadores, o serviço de backend recuperará as informações de conexão e as fornecerá ao cliente de jogo.

Serviços externos

O jogo pode depender de serviços externos, como para validar uma associação por assinatura. Um serviço externo pode passar informações para seus servidores de jogos por meio de um serviço de back-end e. Amazon GameLift Servers

Servidores de jogos

Um servidor de jogos é o software de servidor do jogo executado em um conjunto de recursos de hospedagem. Você carrega o software do servidor de jogos para Amazon GameLift Servers, que o implanta nos recursos de hospedagem e começa a executar os processos do servidor. Cada processo do servidor de jogos se conecta ao Amazon GameLift Servers serviço para sinalizar a prontidão para hospedar sessões de jogo. Ele interage com o serviço para iniciar sessões de jogo, validar jogadores recém-conectados e relatar o status das sessões de jogo e das conexões dos jogadores.

Servidores de jogos personalizados se comunicam Amazon GameLift Servers usando o SDK do servidor para Amazon GameLift Servers. Para obter mais informações, consulte [Integre jogos com servidores de jogos personalizados](#). Realtieservidores são servidores de jogos fornecidos por Amazon GameLift Servers. Você pode personalizar a lógica do servidor fornecendo um script personalizado. Para obter mais informações, consulte [Integrando jogos com Amazon GameLift ServersRealtime](#).

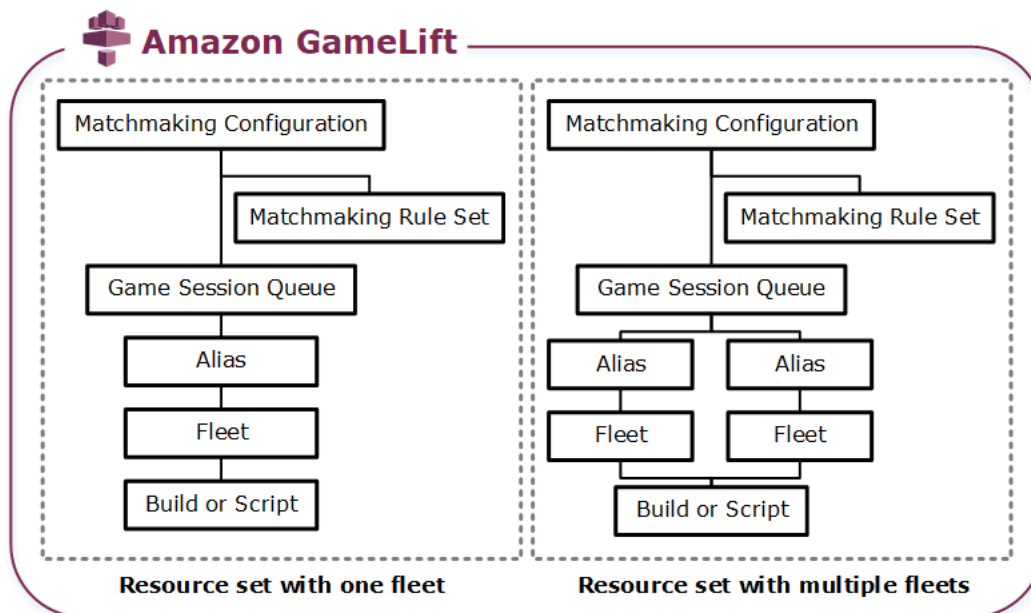
Ferramentas de gerenciamento de host

Ao configurar e gerenciar recursos de hospedagem, os proprietários de jogos usam ferramentas de gerenciamento de hospedagem para gerenciar compilações ou scripts de servidores de

jogos, frotas, matchmaking e filas. O conjunto de Amazon GameLift Servers ferramentas no AWS SDK e no console fornece várias maneiras de gerenciar seus recursos de hospedagem. Acesse remotamente qualquer servidor de jogo individual para solução de problemas.

Recursos da solução de hospedagem

O diagrama a seguir ilustra Amazon GameLift Servers os recursos que compõem uma solução de hospedagem gerenciada. Forneça uma criação de servidor personalizada ou um Amazon GameLift Servers Realtime script, implante uma frota de computadores para hospedar servidores de jogos e, em seguida, configure uma fila de sessões de jogo para encontrar recursos de hospedagem disponíveis e iniciar novas sessões de jogo. Para jogos que usam FlexMatch matchmaking, adicione uma configuração de matchmaking e um conjunto de regras de matchmaking para gerar partidas de jogadores.



Código do servidor de jogos

- **Build** — Seu software de servidor de jogos personalizado que roda Amazon GameLift Servers e hospeda sessões de jogo para seus jogadores. Uma compilação de jogo representa o conjunto de arquivos que executam seu servidor de jogos em um sistema operacional específico e com os quais você deve se integrar Amazon GameLift Servers. Faça upload dos arquivos de compilação do jogo para as Regiões da AWS local Amazon GameLift Servers em que você planeja configurar frotas. Para obter mais informações, consulte [Implemente uma criação de servidor personalizada para Amazon GameLift Servers hospedagem](#).

- **Script** — Sua configuração e lógica de jogo personalizada para usar com Amazon GameLift ServersRealtime. Configure Amazon GameLift Servers Realtime para seus clientes de jogo criando um script usando JavaScript, e adicione uma lógica de jogo personalizada para hospedar sessões de jogo para seus jogadores. Para obter mais informações, consulte [Implante um script para Amazon GameLift ServersRealtime](#).

Frota

Uma coleção de recursos de computação que executa os servidores de jogos e hospeda sessões de jogos para os jogadores. Para obter informações sobre onde é possível implantar frotas, consulte [Amazon GameLift Servers locais de serviço](#). Para obter informações sobre como criar frotas, consulte [Configurando uma frota de hospedagem com Amazon GameLift Servers](#).

Alias

Um identificador abstrato de uma frota que você pode usar para alterar a frota à qual os jogadores estão conectados a qualquer momento. Para obter mais informações, consulte [Crie um Amazon GameLift Servers alias](#).

Fila de sessões de jogo

Um mecanismo de posicionamento de sessões de jogo que recebe solicitações de novas sessões de jogo e procura servidores de jogos disponíveis para hospedar as novas sessões. Para obter mais informações sobre filas de sessões de jogo e como elas funcionam, consulte [Gerenciando o posicionamento da sessão de jogo com Amazon GameLift Servers filas](#).

Como funciona o Amazon GameLift Servers

Este tópico descreve como Amazon GameLift Servers gerencia a hospedagem dedicada para seus servidores de jogos multijogador e os disponibiliza aos jogadores. Ele descreve como os principais recursos funcionam.

Hospedar servidores de jogos

Com Amazon GameLift Servers, você pode hospedar seus servidores de jogos de várias maneiras diferentes: Amazon GameLift Servers gerenciados Amazon GameLift Servers FleetIQ e Amazon GameLift Servers em qualquer lugar. Para obter mais informações sobre Amazon GameLift Servers FleetIQ, consulte [O que é Amazon GameLift Servers FleetIQ?](#)

Você pode projetar uma frota de acordo com as necessidades do seu jogo. Para obter mais informações sobre como projetar uma frota, consulte [Personalize seu Amazon GameLift Servers EC2 frotas gerenciadas](#).

Amazon GameLift Servers gerenciado

Com o gerenciado Amazon GameLift Servers, você pode hospedar seus servidores de jogos em recursos de computação Amazon GameLift Servers virtual, chamados de instâncias. Configure seus recursos de hospedagem criando uma frota de instâncias e implantando-as para executar os servidores de jogos.

Amazon GameLift Servers Em qualquer lugar

Com o Amazon GameLift Servers Anywhere, você pode hospedar seus servidores de jogos no computador que você gerencia. Configure seus recursos de hospedagem criando uma frota Anywhere que faça referência à sua computação.

Aliases de frota

Um alias é uma designação que você poderá transferir entre frotas, tornando-se uma maneira conveniente de ter uma localização genérica da frota. Use um alias para alternar as frotas usadas pelos clientes do jogo sem precisar alterar o cliente do jogo. Também é possível criar um alias de terminal que você aponta para o conteúdo.

Execução de sessões de jogos

Depois de implantar sua compilação de servidor de jogos em uma frota e Amazon GameLift Servers iniciar os processos do servidor de jogos em cada instância, a frota poderá hospedar sessões de jogo. Amazon GameLift Servers inicia novas sessões de jogo quando seu serviço de cliente de jogos envia uma solicitação de posicionamento para o serviço de back-end ou para o Amazon GameLift Servers.

Posicionamento da sessão de jogo e o FleetIQ algoritmo

As filas usam o FleetIQ algoritmo para selecionar um servidor de jogo disponível para hospedar uma nova sessão de jogo. O principal componente para o posicionamento da sessão de jogo é a fila da sessão de Amazon GameLift Servers jogo. Você atribui a uma fila de sessões de jogo uma lista de frotas, que determina onde a fila pode colocar as sessões de jogo. Para obter mais informações sobre filas de sessões de jogo e como criá-las para seu jogo, consulte [Personalize uma fila de sessões de jogo](#).

Otimizando o posicionamento da sessão de jogo com sinalizadores de ping UDP

Usando beacons de ping Amazon GameLift Servers UDP, você pode calcular a latência de ida e volta para pacotes UDP entre jogadores e servidores de jogos em locais diferentes para ajudar a escolher o local ideal para uma sessão de jogo. Para obter mais informações sobre os beacons de ping UDP e como você pode usá-los para medir a latência, consulte [Sinalizadores de ping UDP](#)

- Para jogos que usam filas de sessão de jogo para posicionamento, a solicitação de posicionamento pode incluir dados de latência, que a fila usa automaticamente para priorizar os locais de posicionamento. Você pode personalizar ainda mais a priorização e configurar políticas, incluindo limites de valores de latência. Consulte [Priorize o posicionamento de sessões de jogo](#).
- Para jogos que não usam filas de sessão de jogo, mas têm frotas com vários locais, você pode avaliar os dados de latência e escolher o melhor local disponível antes de fazer uma solicitação de sessão de jogo para. Amazon GameLift Servers Veja as seções Obter e Criar sessões de jogo em [Adicione Amazon GameLift Servers ao seu cliente de jogo](#).
- Se você estiver usando FlexMatch para matchmaking, você pode definir regras de partida para usar dados de latência. Consulte [Solicitar matchmaking para jogadores](#) e a seção [Regra de latência](#) dos tipos de FlexMatch regras no Guia do Amazon GameLift Servers FlexMatch desenvolvedor.

Conexões de jogadores com jogos

Como parte do processo de colocação da sessão de jogo, a fila ou sessão de jogo solicita que o servidor de jogo selecionado inicie uma nova sessão de jogo. O servidor do jogo responde à solicitação e informa Amazon GameLift Servers quando está pronto para aceitar as conexões dos jogadores. Amazon GameLift Servers em seguida, entrega as informações de conexão ao serviço de back-end ou ao serviço de cliente do jogo. Seus clientes de jogo usam essas informações para se conectarem diretamente à sessão de jogo e iniciar o jogo.

Escalar a capacidade de frota

Quando a frota estiver ativa e pronta para hospedar sessões de jogos, poderá ajustar a capacidade da frota para atender à demanda dos jogadores. É recomendável buscar um equilíbrio entre todos os novos jogadores encontrarem um jogo rapidamente e gastarem demais em recursos que ficam ociosos.

Amazon GameLift Servers fornece uma ferramenta de escalonamento automático altamente eficaz, ou você pode definir manualmente a capacidade da frota. Para obter mais informações, consulte [Escalando a capacidade de hospedagem de jogos com Amazon GameLift Servers](#).

Ajuste de escala automático

Amazon GameLift Servers fornece dois métodos de escalonamento automático:

- [Ajuste de escala automático baseado em destino](#)
- [Ajuste de escala automático com políticas com base nas regras](#)

Recursos de escalabilidade adicionais

- Proteção de sessão de jogo — Evite encerrar sessões Amazon GameLift Servers de jogo que estejam hospedando jogadores ativos durante um evento de redução de escala.
- Limites de escalabilidade – Controla uso geral de instâncias definindo limites mínimos e máximos para o número de instâncias em uma frota.
- Suspensão do ajuste de escala automático – Suspenda o ajuste de escala automático no nível da localização da frota sem alterar ou excluir suas políticas de ajuste de escala automático.
- Métricas de escalabilidade – Rastreie o histórico de capacidade e eventos de escalabilidade de uma frota.

Como monitorar o Amazon GameLift Servers

Quando você tem frotas instaladas e funcionando, Amazon GameLift Servers coleta uma variedade de informações para ajudá-lo a monitorar o desempenho dos servidores de jogos implantados. Você pode usar essas informações para otimizar o uso de recursos, solucionar problemas e obter informações sobre como os jogadores estão ativos em seus jogos. Amazon GameLift Servers coleta o seguinte:

- Detalhes da frota, localização, sessão de jogo e sessão do jogador
- Métricas de uso
- Integridade do processo do servidor
- Logs de sessão de jogo

Para obter mais informações sobre o monitoramento em Amazon GameLift Servers, consulte [Monitoramento Amazon GameLift Servers](#).

Usando outros AWS recursos

Seus servidores e aplicativos de jogos podem se comunicar com outros AWS recursos. Por exemplo, use um conjunto de serviços da Web para autenticação de jogador ou redes sociais. Para que seus servidores de jogos acessem AWS os recursos que você Conta da AWS gerencia, permita explicitamente Amazon GameLift Servers o acesso aos seus AWS recursos.

O Amazon GameLift Servers fornece algumas opções de gerenciamento desse tipo de acesso. Para obter mais informações, consulte [Comunique-se com outros AWS recursos de suas frotas](#).

Como os contêineres funcionam em Amazon GameLift Servers

Amazon GameLift Servers as frotas de contêineres são projetadas para oferecer flexibilidade na forma como você implanta e dimensiona seus aplicativos em contêineres. Ele usa o Amazon Elastic Container Service (Amazon ECS) para gerenciar a implantação e execução de tarefas para suas frotas. Este tópico descreve os elementos estruturais básicos para a execução de contêineres em uma frota Amazon GameLift Servers gerenciada, ilustra arquiteturas comuns e descreve alguns conceitos principais.

 Acelere a integração com essas ferramentas para contêineres gerenciados:

- O [kit inicial de contêineres](#) agiliza a integração e a configuração da frota. Ele adiciona recursos essenciais de gerenciamento de sessões de jogo ao seu servidor de jogos e usa modelos pré-configurados para criar uma frota de contêineres e um pipeline de implantação automatizado para seu servidor de jogos. Após a implantação, use o Amazon GameLift Servers console e as ferramentas de API para monitorar o desempenho da frota, gerenciar sessões de jogo e analisar métricas.
- Para desenvolvedores do Unreal Engine ou Unity, use os [Amazon GameLift Servers plug-ins](#) para integrar seu servidor de jogos e criar uma frota de contêineres de dentro do ambiente de desenvolvimento do seu mecanismo de jogo. Os fluxos de trabalho guiados do plug-in ajudam você a criar uma solução rápida e simples com hospedagem baseada em nuvem usando contêineres gerenciados. Em seguida, desenvolva essa base para criar uma solução de hospedagem personalizada para seu jogo.

componentes da frota de contêineres

Frota

Uma frota de contêineres é uma coleção de EC2 instâncias da Amazon para hospedar seus servidores de jogos em contêineres. Essas instâncias são gerenciadas Amazon GameLift Servers em seu nome. Ao criar uma frota, você configura como a arquitetura de contêiner com seu software de servidor de jogos é implantada em cada instância da frota. Você pode criar uma frota de contêineres com instâncias em uma ou várias localizações geográficas. Você pode usar ferramentas Amazon GameLift Servers de escalabilidade para escalar automaticamente a capacidade de uma frota de contêineres para hospedar sessões de jogos e jogadores.

Instância

Uma EC2 instância da Amazon é o servidor virtual que fornece capacidade computacional para hospedagem de jogos. Com Amazon GameLift Servers, você pode escolher entre uma variedade de tipos de instância. Cada tipo de instância oferece uma combinação diferente de CPU, memória, armazenamento e capacidade de rede.

Quando você cria uma frota de contêineres, Amazon GameLift Servers implanta seus contêineres com base no tipo de instância escolhido e na configuração da sua frota. Cada instância de frota implantada é idêntica e executa seu software de servidor de jogos em contêineres da mesma forma. O número de instâncias em uma frota determina o tamanho da frota e a capacidade de hospedagem de jogos.

Grupo de contêineres

Amazon GameLift Servers usa o conceito de um grupo de contêineres para descrever e gerenciar um conjunto de contêineres. Um grupo de contêineres é semelhante a uma “tarefa” ou “pod” de contêineres. Em cada grupo de contêineres, você pode definir como os contêineres se comportam, configurar dependências e compartilhar os recursos de CPU e memória disponíveis.

Cada instância da frota pode ter os seguintes tipos de grupos de contêineres:

- Um grupo de contêineres de servidores de jogos gerencia os contêineres que executam seu aplicativo de servidor de jogos e o software de suporte. Uma frota de contêineres deve ter um desse tipo de grupo de contêineres para hospedar sessões de jogo e jogadores. Um grupo de contêineres de servidores de jogos pode ser replicado em uma instância de frota. O número de réplicas de grupos de servidores de jogos por instância da frota depende dos requisitos de computação do seu software e dos recursos computacionais disponíveis na instância.

- Um grupo de contêineres por instância, que é opcional, oferece a capacidade de executar software adicional em cada instância da frota. Eles são úteis para executar serviços em segundo plano ou programas utilitários, como para monitoramento. Seu software de servidor de jogos não depende diretamente dos processos em um grupo por instância. Somente uma cópia de um grupo de contêineres por instância é implantada em cada instância da frota.

Cada grupo de contêineres em uma frota de contêineres tem um contêiner designado como “essencial”. Um contêiner essencial impulsiona o ciclo de vida de um grupo de contêineres. Se o contêiner essencial falhar, todo o grupo de contêineres será reiniciado.

Contêiner

O contêiner é o elemento mais básico de uma arquitetura baseada em contêiner. Ele inclui uma imagem de contêiner com executáveis de software e arquivos dependentes. Defina um contêiner para configurar como o software é executado e com Amazon GameLift Servers o qual interage.

Amazon GameLift Servers define dois tipos de contêineres:

- Um contêiner de servidor de jogos inclui tudo o que você precisa para executar os processos do servidor de jogos e hospedar sessões de jogo para os jogadores. Inclui a versão do seu servidor de jogo e o software dependente. Defina um contêiner de servidor de jogos para o grupo de contêineres de servidores de jogos de uma frota. O contêiner do servidor do jogo é automaticamente considerado essencial para o grupo de contêineres.
- Um contêiner de suporte executa software adicional para dar suporte ao seu servidor de jogos. É semelhante ao conceito de contêiner “sidecar”. Ele oferece a opção de executar e escalar o software de suporte junto com seus servidores de jogos, mas gerenciá-lo como contêineres separados. Em um grupo de contêineres de servidores de jogos, você pode definir zero ou mais contêineres de suporte. Em um grupo de contêineres por instância, todos os contêineres são contêineres de suporte. Qualquer contêiner de suporte pode ser designado como essencial.

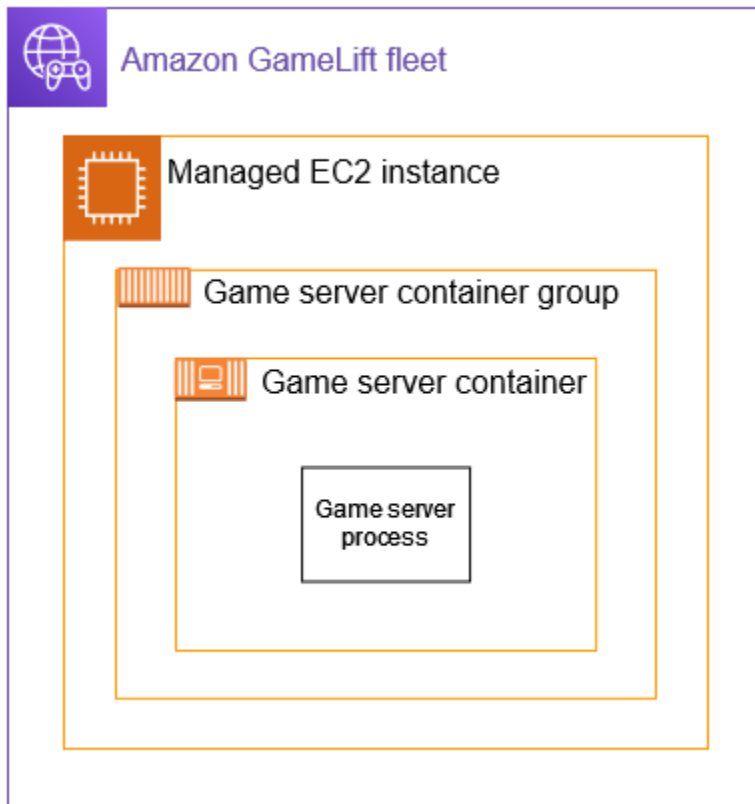
Computação

Uma computação representa uma cópia de um grupo de contêineres de servidores de jogos em uma instância de frota.

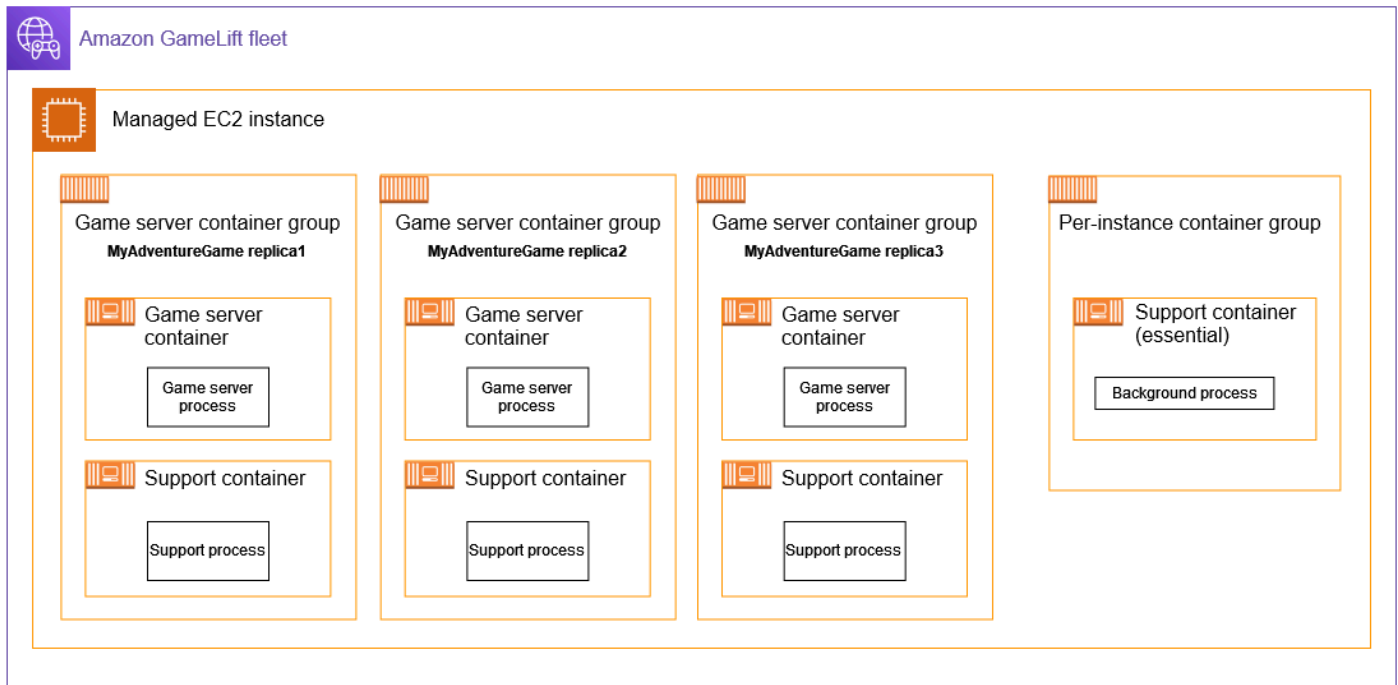
Arquiteturas comuns

O diagrama a seguir ilustra a estrutura mais simples da frota de contêineres. Nessa estrutura, cada instância da frota mantém uma cópia do grupo de contêineres do servidor do jogo. O grupo de

contêineres tem um único contêiner de servidor de jogos que executa um processo de servidor de jogos. Neste exemplo, a frota de contêineres está configurada para colocar uma cópia do grupo de contêineres do servidor de jogos por instância. Com essa arquitetura, cada instância executa um processo de servidor de jogos.



Esse segundo diagrama ilustra uma arquitetura de frota de contêineres mais complexa. Nessa estrutura, a frota tem um grupo de contêineres de servidores de jogos e um grupo de contêineres por instância. O grupo de contêineres do servidor de jogos tem contêineres separados para o processo do servidor de jogos e um processo de suporte. A frota está configurada para colocar três cópias do grupo de contêineres do servidor do jogo em cada instância da frota. O grupo de contêineres por instância nunca é replicado. Neste exemplo, a frota de contêineres está configurada para colocar três cópias do grupo de contêineres do servidor de jogos por instância. Com essa arquitetura, cada instância executa três processos de servidor de jogos.



Atributos principais

Esta seção resume como Amazon GameLift Servers implementa alguns conceitos básicos de contêiner. Para obter instruções sobre como trabalhar com frotas de contêineres, consulte os tópicos relevantes neste guia.

Atualizações ativas da frota

Os contêineres gerenciados fornecem suporte avançado para ajudá-lo a gerenciar o ciclo de vida do seu software hospedado e da arquitetura de contêineres. Você pode atualizar suas definições de contêiner, incluindo imagens de contêiner, e implantar alterações em suas frotas existentes. Esse recurso torna mais rápido e fácil iterar as alterações em seus contêineres durante o desenvolvimento. Ele também fornece recursos para ajudá-lo a criar, implantar e monitorar suas atualizações de versão de software ao longo do tempo. Esses recursos incluem:

- Gerencie as atualizações e o controle de versão das definições de grupos de contêineres. Você pode atualizar quase todas as propriedades de uma definição de grupo de contêineres, incluindo imagens de contêineres e definições de configuração. Sempre que você atualiza um contêiner, atribui Amazon GameLift Servers automaticamente um número de versão à atualização e, por padrão, mantém todas as versões. Você pode acessar qualquer versão específica e excluir as versões conforme desejado. Ao criar uma frota de contêineres, você pode especificar a definição e a versão do grupo de contêineres a ser implantada.

- Atualize as frotas de contêineres existentes com novas definições e configurações de grupos de contêineres. Você pode implantar atualizações de contêineres em frotas que já estão implantadas em instâncias de frota. Você pode acompanhar o status das implantações de atualização em cada local da frota usando o AWS Management Console, o AWS SDK e a CLI.
- Configure como você deseja que as atualizações da frota sejam implantadas em uma frota ativa.
 - Proteção da sessão do jogo. Escolha proteger as instâncias da frota com sessões de jogo ativas até o final das sessões de jogo (implantação segura). Ou opte por substituir as instâncias da frota, independentemente da atividade da sessão de jogo (implantação insegura). Use implantações inseguras durante as fases de desenvolvimento e teste para reduzir o tempo de implantação.
 - Porcentagem mínima saudável. Especifique a porcentagem de tarefas íntegras que você deseja manter durante a implantação. Esse recurso permite que você decida quantas instâncias da frota são afetadas durante uma implantação. Um valor baixo prioriza a velocidade de implantação, enquanto um valor alto garante que a disponibilidade do servidor de jogos permaneça alta durante toda a implantação.
 - Estratégia de falha na implantação. Decida quais ações tomar se uma implantação falhar. Uma falha na implantação significa que alguns dos contêineres atualizados falharam nas verificações de status e são considerados danificados. Você pode definir implantações para reverter automaticamente todas as instâncias da frota para o estado implantado anteriormente. Como alternativa, você pode optar por manter algumas das instâncias da frota danificadas para uso na depuração.

A capacidade de atualizar frotas ativas é muito útil quando você deseja implantar uma atualização no software do servidor de jogos. Depois de criar uma nova imagem de contêiner para seu servidor de jogos, implantá-la é um processo de duas etapas: primeiro, atualizar a definição do grupo de contêineres com a nova imagem e, segundo, atualizar a frota de contêineres. Amazon GameLift Servers lida com todas as outras tarefas conforme necessário.

Embalagem de recipientes

Ao desenvolver sua estrutura de contêineres para implantação em uma frota de contêineres, um objetivo comum é otimizar o uso da capacidade de computação disponível. Para atingir esse objetivo, você deve incluir o maior número possível de grupos de contêineres de servidores de jogos em cada instância da frota.

Amazon GameLift Servers ajuda você a fazer isso calculando o máximo de grupos de contêineres de servidores de jogos por instância, com base nas seguintes informações:

- O tipo de instância da frota e seus recursos de vCPU e memória.
- Os requisitos de vCPU e memória para todos os contêineres no grupo de contêineres do servidor de jogos.

Os requisitos de vCPU e memória para todos os contêineres em um grupo de contêineres por instância, se houver um.

Ao criar uma frota de contêineres, você pode usar o máximo calculado ou especificar o número desejado. Como prática recomendada, planeje fazer experiências com seu software de servidor de jogos em contêineres para determinar os requisitos de recursos para um desempenho ideal do servidor de jogos.

Escalabilidade da capacidade

A capacidade da frota mede o número de sessões de jogo que uma frota pode hospedar simultaneamente. Você também pode medir a capacidade com base no número de jogadores simultâneos que uma frota pode suportar. Para aumentar ou diminuir a capacidade de hospedagem de uma frota, você adiciona ou remove instâncias da frota.

Uma frota de contêineres é configurada para executar um número específico de processos simultâneos do servidor de jogos em cada instância da frota. (Você pode calcular isso com base em (1) os grupos de contêineres do servidor de jogos por instância e (2) o número de processos do servidor de jogos que são executados em cada grupo de contêineres.) O número de servidores de jogos simultâneos por instância indica qual é o impacto de adicionar ou remover cada instância da frota. Por exemplo, se sua frota de contêineres executa 1 processo de servidor de jogo em cada grupo de contêineres de servidores de jogos e cada instância da frota contém 100 grupos de contêineres de servidores de jogos, você aumenta ou diminui a capacidade da sua frota de hospedar sessões de jogo simultâneas em incrementos de 100. Se cada sessão de jogo tiver 10 vagas para jogadores, você aumentará ou diminuirá a capacidade da sua frota de hospedar jogadores em incrementos de 1000.

Com frotas de contêineres, você pode usar qualquer um dos métodos de escalonamento de capacidade fornecidos pela Amazon GameLift Servers. Isso inclui:

- Defina a capacidade da frota manualmente definindo a contagem de instâncias da frota desejada.
- Configure o escalonamento automático visando o buffer desejado de instâncias disponíveis (rastreamento de metas). Esse método mantém automaticamente uma certa quantidade de recursos de hospedagem ociosos para que os novos jogadores possam entrar nos jogos

rapidamente. Conforme a demanda do jogador aumenta ou diminui, o tamanho desse buffer é ajustado continuamente.

- Configure o escalonamento automático com regras de escalabilidade personalizadas (recurso avançado). Esse método permite que você escale com base nas métricas de frota que você escolher.

client/server Conexões de jogos

Com frotas Amazon GameLift Servers gerenciadas, os clientes de jogos se conectam diretamente aos seus servidores de jogos hospedados na nuvem. Quando um cliente de jogo pede para entrar em um jogo, Amazon GameLift Servers encontra uma sessão de jogo e fornece informações de conexão (IP e porta) ao cliente do jogo. Você pode controlar o acesso externo às instâncias da frota abrindo determinados intervalos de portas (permissões de entrada) para a frota. As permissões de entrada determinam quais portas estão abertas para o tráfego de entrada. Você pode desligar rapidamente todas as portas, limitar a algumas ou abrir todas as portas.

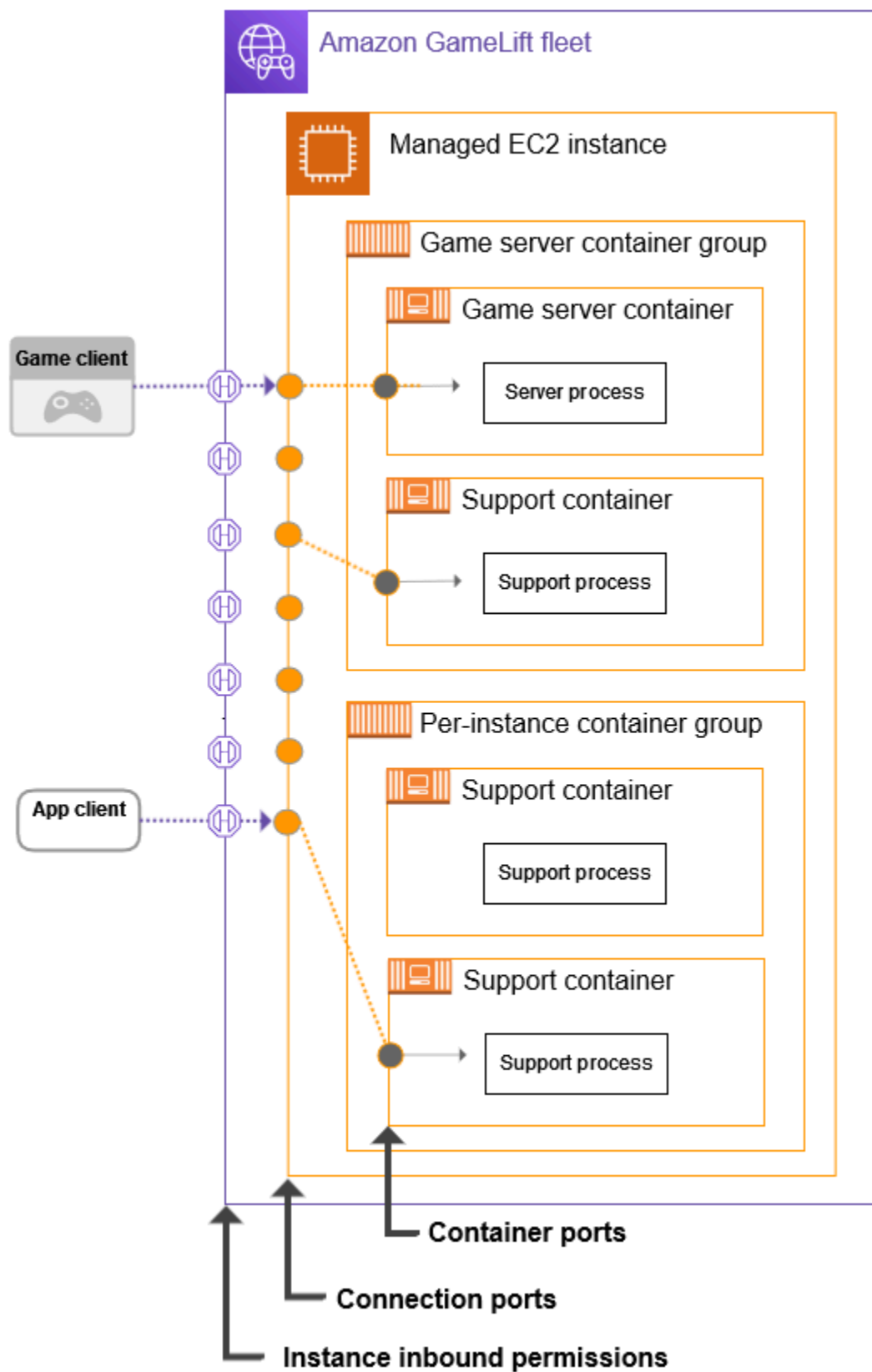
As frotas de contêineres gerenciados exigem uma configuração adicional que permita o acesso aos processos que estão sendo executados em um contêiner. Ao criar uma definição de contêiner, você especifica um conjunto de portas, uma para cada processo que usa uma conexão. Isso inclui:

- Todos os processos do servidor de jogos que serão executados simultaneamente no contêiner do servidor de jogos. Todos os processos do servidor de jogos devem permitir que os clientes do jogo se conectem para participar de uma sessão de jogo.
- Qualquer processo em um contêiner de suporte ao qual uma fonte externa precise se conectar. Por exemplo, você pode se conectar remotamente a um aplicativo de teste.

Ao definir as configurações de porta de contêiner voltadas para o interior, Amazon GameLift Servers use-as para calcular as permissões de entrada externas às quais os clientes do jogo e outros aplicativos podem se conectar. Amazon GameLift Servers também gerencia o mapeamento entre permissões de entrada e portas de contêiner individuais que dá aos jogadores acesso a uma sessão de jogo em um contêiner. Esse mapeamento interno fornece uma camada de segurança ao proteger seus servidores de jogos do acesso direto às portas do contêiner. Você tem a opção de personalizar as configurações de portas externas de uma frota, conforme necessário. Para obter mais informações sobre como configurar manualmente as portas da frota de contêineres, consulte [Configurar conexões de rede](#).

Você pode modificar as configurações portuárias de uma frota de contêineres a qualquer momento. Essa alteração exige a implantação de uma atualização de frota.

O diagrama a seguir ilustra o papel das conexões portuárias em uma frota de contêineres. Conforme mostrado, você define portas em contêineres individuais e Amazon GameLift Servers usa essas informações para configurar portas suficientes na instância da frota para mapear para cada porta de contêiner. Tanto as permissões de entrada da instância voltada para a externa quanto as portas de conexão internas são calculadas para sua frota, a menos que você Amazon GameLift Servers opte por configurá-las manualmente.



Registro de contêiner

Em frotas de contêineres gerenciadas, os fluxos de saída padrão (e erro padrão) são capturados para todos os contêineres. Isso inclui os registros de sessão de jogo do seu servidor de jogo. Você pode configurar uma frota de contêineres para usar uma das várias opções para lidar com fluxos de saída:

- Salve a saída do contêiner como um fluxo de CloudWatch log da Amazon. Cada fluxo de log faz referência ao ID da frota e ao contêiner. Se você escolher essa opção de registro para a frota, você especifica um grupo de CloudWatch registros, que organiza todos os fluxos de registro da frota. Em seguida, você pode usar os CloudWatch recursos para pesquisar e analisar os dados de registro conforme necessário.
- Salve a saída do contêiner em um bucket de armazenamento do Amazon Simple Storage Service (Amazon S3). Você pode visualizar, compartilhar ou baixar o conteúdo conforme necessário.
- Desative o registro. Nesse cenário, a saída do contêiner não é salva.

Amazon GameLift Servers envia dados de log de frotas de contêineres gerenciados para os serviços do Amazon S3 CloudWatch ou do Amazon S3 em AWS sua conta. Para visualizar seus dados, use as AWS Management Console ou outras ferramentas fazendo login em sua AWS conta e trabalhando com os serviços individuais. Você estende o acesso limitado Amazon GameLift Servers para realizar essas ações criando uma função de serviço para suas frotas de contêineres.

Você pode modificar a configuração de registro de uma frota de contêineres a qualquer momento. Essa alteração exige a implantação de uma atualização de frota.

Frotas de contêineres e o agente Amazon GameLift Servers

Uma arquitetura de contêiner comumente usada executa um único processo em cada contêiner. Em uma frota de Amazon GameLift Servers contêineres, o grupo de contêineres de servidores de jogos tem um contêiner de servidor de jogos, que executa um processo de servidor de jogos. Com essa arquitetura, Amazon GameLift Servers gerencia o ciclo de vida do processo de um único servidor de jogo em cada grupo de contêineres de servidores de jogos em uma instância de frota.

Se você optar por criar uma arquitetura de contêiner que execute vários processos de servidor de jogos em cada grupo de contêineres de servidores de jogos, precisará de uma forma de gerenciar o ciclo de vida de todos os processos. Isso inclui tarefas como iniciar, encerrar e substituir processos conforme necessário, gerenciar o número desejado de processos a serem executados simultaneamente e lidar com estados de falha.

Você pode optar por usar o Amazon GameLift Servers Agente para essas tarefas. Para uma frota de contêineres, o Agente implementa instruções de tempo de execução que especificam quais executáveis executar (e quantos), fornecem parâmetros de inicialização e definem regras para a ativação do servidor de jogos. Por exemplo, instruções de tempo de execução podem instruir o agente a manter dez processos de servidor de jogos para uso em produção e um processo de servidor de jogos com parâmetros de inicialização especiais para teste.

Para usar o Agente com suas frotas de contêineres, adicione o Agente à imagem do contêiner e inclua um conjunto de instruções de tempo de execução. Para obter mais informações sobre o Agente, consulte [Trabalhe com o Amazon GameLift Servers agente](#).

Como os jogadores se conectam aos jogos

Uma sessão de jogo é uma instância do jogo em execução no Amazon GameLift Servers. Para jogar o jogo, um jogador pode encontrar e ingressar em uma sessão de jogo existente ou criar uma sessão de jogo nova e adicioná-la. Um jogador ingressa criando uma sessão de jogador para a sessão de jogo. Se a sessão do jogo estiver aberta para jogadores, Amazon GameLift Servers reserve um espaço para o jogador e forneça informações de conexão. O jogador pode acabar se conectando à sessão de jogo e reivindicando o slot reservado.

Para obter informações detalhadas sobre como criar e gerenciar sessões de jogos e sessões de jogadores com servidores de jogos personalizados, consulte [Adicione Amazon GameLift Servers ao seu cliente de jogo](#). Para obter informações sobre como conectar jogadores ao Amazon GameLift ServersRealtime, consulte [Integrando um cliente de jogo para Amazon GameLift ServersRealtime](#).

Amazon GameLift Servers fornece vários recursos relacionados às sessões de jogos e jogadores.

Hospedar sessões de jogo nos melhores recursos disponíveis entre vários locais

Escolha uma dentre várias opções ao configurar como o Amazon GameLift Servers seleciona recursos para hospedar novas sessões de jogo. Caso esteja administrando frotas em vários locais, poderá criar filas de sessões de jogo que coloquem novas sessões de jogo em qualquer frota, independentemente da localização. Use beacons de ping UDP para calcular os dados de latência, que a fila usa automaticamente para priorizar os locais de posicionamento.

Controlar acesso do jogador a sessões de jogos

Configure sessões de jogo para permitir ou negar solicitações de entrada de novos jogadores, independentemente do número de jogadores conectados.

Usar dados de jogo e jogador personalizados

Adicione dados personalizados aos objetos da sessão do jogo e aos objetos da sessão do jogador. Amazon GameLift Servers passa os dados da sessão de jogo para um servidor de jogo ao iniciar uma nova sessão de jogo. Amazon GameLift Servers passa os dados da sessão do jogador para o servidor do jogo quando um jogador se conecta à sessão do jogo.

Filtrar e classificar sessões de jogos disponíveis

Use a pesquisa e classificação da sessão para encontrar a melhor correspondência possível para um jogador em potencial ou permita o jogador escolher em uma lista de sessões de jogo disponíveis. Use a pesquisa e classificação de sessões para encontrar sessões de jogo com base nas características da sessão. Também é possível pesquisar e classificar de acordo com seus próprios dados personalizados do jogo.

Acompanhar dados de uso de jogo e jogador

Armazene automaticamente registros de sessões de jogo concluídas. Você pode configurar o armazenamento de registros ao se Amazon GameLift Servers integrar aos seus servidores de jogos. Para obter mais informações, consulte [Registrando mensagens do servidor em Amazon GameLift Servers](#).

Use o Amazon GameLift Servers console para ver informações detalhadas sobre as sessões do jogo, incluindo metadados da sessão, configurações e dados da sessão do jogador. Para obter mais informações, consulte [Sessões de jogos e jogadores no Amazon GameLift Servers console](#) e [Métricas](#).

Amazon GameLift Servers locais de serviço

Amazon GameLift Servers recursos estão disponíveis em várias Regiões da AWS Zonas Locais. Você pode criar uma solução de hospedagem que coloque seus servidores de jogos exatamente onde seus jogadores estão localizados.

AWS Localizações suportadas

A tabela a seguir é uma lista de Zonas Regiões da AWS Locais que oferecem suporte a Amazon GameLift Servers recursos. Ela indica os tipos de recursos que você pode criar em cada local.

Localização geográfica	Código de localização	Região de origem para frotas gerenciadas (local único)	Região de origem para frotas gerenciadas (vários locais)	Localização remota para frotas gerenciadas (vários locais)	Frota em qualquer lugar	Fila de sessões de jogo	FlexMatch matchmaking e conjunto de regras
Leste dos EUA (Norte da Virgínia)	us-east-1	Sim	Sim	Sim	Sim	Sim	Sim
Leste dos EUA (Ohio)	us-east-2	Sim		Sim	Sim	Sim	
Oeste dos EUA (N. da Califórnia)	us-west-1	Sim		Sim	Sim	Sim	
Oeste dos EUA (Oregon)	us-west-2	Sim	Sim	Sim	Sim	Sim	Sim
África (Cidade do Cabo)	af-south-1			Sim			
Ásia-Pacífico (Tailândia)	ap-southeast-7			Sim			
Ásia-Pacífico (Hong Kong)	ap-east-1			Sim			
Ásia-Pacífico (Malásia)	ap-southeast-5			Sim			

Localização geográfica	Código de localização	Região de origem para frotas gerenciadas (local único)	Região de origem para frotas gerenciadas (vários locais)	Localização remota para frotas gerenciadas (vários locais)	Frota em qualquer lugar	Fila de sessões de jogo	FlexMatch matchmaking e conjunto de regras
Ásia-Pacífico (Mumbai)	ap-south-1	Sim		Sim	Sim	Sim	
Ásia Pacífico (Osaka)	ap-northeast-3			Sim			
Ásia-Pacífico (Seul)	ap-northeast-2	Sim	Sim	Sim	Sim	Sim	Sim
Ásia-Pacífico (Singapura)	ap-southeast-1	Sim		Sim	Sim	Sim	
Ásia-Pacífico (Sydney)	ap-southeast-2	Sim	Sim	Sim	Sim	Sim	Sim
Ásia-Pacífico (Tóquio)	ap-northeast-1	Sim	Sim	Sim	Sim	Sim	Sim
Canadá (Central)	ca-central-1	Sim		Sim	Sim	Sim	
Europa (Frankfurt)	eu-central-1	Sim	Sim	Sim	Sim	Sim	Sim

Localização geográfica	Código de localização	Região de origem para frotas gerenciadas (local único)	Região de origem para frotas gerenciadas (vários locais)	Localização remota para frotas gerenciadas (vários locais)	Frota em qualquer lugar	Fila de sessões de jogo	FlexMatch matchmaking e conjunto de regras
Europa (Irlanda)	eu-west-1	Sim	Sim	Sim	Sim	Sim	Sim
Europa (Londres)	eu-west-2	Sim		Sim	Sim	Sim	
Europa (Milão)	eu-south-1			Sim			
Europa (Paris)	eu-west-3			Sim			
Europa (Estocolmo)	eu-north-1			Sim			
Oriente Médio (Barém)	me-south-1			Sim			
América do Sul (São Paulo)	sa-east-1	Sim		Sim	Sim	Sim	
Zona local de Atlanta	us-east-1-atl-1			Sim			
Zona local de Chicago	us-east-1-chi-1			Sim			

Localização geográfica	Código de localização	Região de origem para frotas gerenciadas (local único)	Região de origem para frotas gerenciadas (vários locais)	Localização remota para frotas gerenciadas (vários locais)	Frota em qualquer lugar	Fila de sessões de jogo	FlexMatch matchmaking e conjunto de regras
Zona local de Dallas	us-east-1-dfw-1			Sim			
Zona local de Denver	us-west-2-den-1			Sim			
Zona local de Houston	us-east-1-iah-1			Sim			
Zona local de Kansas City	us-east-1-mci-1			Sim			
Zona local de Los Angeles	us-west-2-lax-1			Sim			
Zona local de Phoenix	us-west-2-phx-1			Sim			
Zona local de Lagos, Nigéria	af-south-1-los-1			Sim			

Note

Nem todas as Regiões da AWS estão habilitadas por padrão para uma Conta da AWS. Se você quiser uma frota de vários locais com instâncias nessas regiões, será necessário habilitá-las. Para obter mais informações sobre regiões que não estão habilitadas por padrão e como habilitá-las, consulte [Gerenciando Regiões da AWS](#) no Referência geral da AWS. As frotas que você criou antes de 28 de fevereiro de 2022 não são afetadas.

Além disso, você deve atualizar sua política de Amazon GameLift Servers administrador para permitir a `ec2:DescribeRegions` ação. Para obter um exemplo de política com regiões que não estão habilitadas por padrão, consulte [Exemplos de permissões administrativas](#).

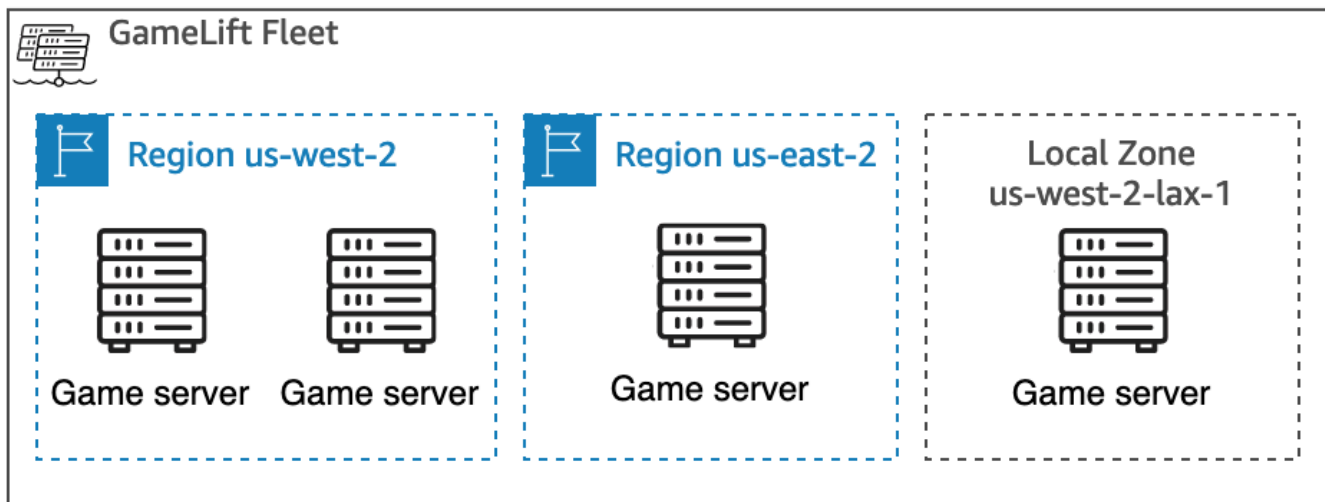
Locais de hospedagem gerenciada

Amazon GameLift Servers hospedagem gerenciada implanta frotas de recursos de servidores de jogos. Cada frota é criada em uma Região da AWS, que é a região de origem da frota. A região de origem de uma frota é referenciada no Amazon Resource Number (ARN) da frota.

Você pode implantar uma frota de uma única região, com recursos de hospedagem somente na região de origem. Como alternativa, é possível implantar uma frota de vários locais, com recursos de hospedagem em várias localizações geográficas. Uma frota com vários locais tem uma região de origem e um ou mais locais remotos. Ao gerenciar a capacidade de hospedagem de uma frota com vários locais, você pode definir a capacidade para cada local individualmente.

Os locais remotos para uma frota com vários locais podem ser Outras Regiões da AWS ou Zonas Locais. Uma zona local é uma extensão de uma Região da AWS. Ele permite que você coloque os recursos de computação mais perto dos usuários para oferecer uma jogabilidade com menor latência. Para obter mais informações, consulte [AWS Local Zones](#). O código de localização de uma zona local é o código da região principal seguido por um identificador de localização física. Por exemplo, o código da zona local de Los Angeles é `us-west-2-lax-1`.

O diagrama a seguir ilustra uma frota de vários locais com recursos em duas Regiões da AWS e uma zona local. A região de origem da frota é `us-west-2`, e ela tem dois locais remotos: `us-east-2` região e zona `us-west-2-lax-1` local.



Além dos recursos da frota, a hospedagem gerenciada com também Amazon GameLift Servers pode usar os seguintes tipos de recursos. Você cria esses recursos em um específico Região da AWS que ofereça suporte ao tipo de recurso.

- **Construir** — Este é um servidor de jogo criado para ser hospedado com uma EC2 frota gerenciada. Crie um recurso de construção na mesma região da frota na qual ele será implantado.
- **Script** — Este é um script de configuração para hospedar um jogo com Amazon GameLift ServersRealtime. Crie um recurso de script na mesma região da frota na qual ele será implantado.
- **Definição do grupo de contêineres e imagem do contêiner** — Essa é uma configuração para executar contêineres em uma frota de contêineres gerenciada. Ele identifica uma ou mais imagens de contêineres com software para implantar na frota de contêineres. Crie uma definição de grupo de contêineres e todas as imagens de contêineres (que são armazenadas em um repositório do Amazon Elastic Container Registry) na mesma região da frota em que serão implantadas.
- **Fila de sessões de jogo** — Esse recurso processa solicitações de sessões de jogo e inicia novas sessões de jogo. O processamento ocorre no local Região da AWS onde a fila está localizada. Para reduzir a latência no processo de posicionamento da sessão de jogo, crie uma fila geograficamente próxima aos jogadores que a usarão.

Localizações para Amazon GameLift Servers qualquer lugar

Uma frota Amazon GameLift Servers Anywhere é uma coleção de hardware de hospedagem que você fornece. Você gerencia todas as atividades em seus recursos de hospedagem, incluindo a implantação do software do servidor de jogos, a manutenção dele atualizado e a inicialização dos

processos do servidor. Você cria uma frota Anywhere para conectar o Amazon GameLift Servers serviço aos seus recursos de hospedagem autogerenciada. Amazon GameLift Servers gerencia o posicionamento da sessão de jogo — processa as solicitações de ingresso de jogadores, localiza os recursos de hospedagem disponíveis, inicia novas sessões de jogo e fornece informações de conexão aos clientes do jogo. Você pode criar uma frota Anywhere em qualquer uma das Regiões da AWS que a suportem.

Você adiciona instâncias de hardware de hospedagem a uma frota Anywhere registrando-a. Cada instância registrada deve ter um local personalizado associado a ela. Os locais personalizados não estão relacionados às Regiões da AWS Zonas Locais. Eles são usados para representar a localização física do hardware.

Para obter mais informações sobre como criar uma frota Anywhere e testar a integração do servidor de jogos, consulte [Crie um Amazon GameLift Servers Frota em qualquer lugar](#) [Configure testes locais com o Amazon GameLift Servers Anywhere](#) e.

Localizações para Amazon GameLift ServersFlexMatch

FlexMatch recursos são usados para processar solicitações de jogadores para matchmaking. Eles incluem um recurso de configuração de matchmaking e um recurso de conjunto de regras. O processamento ocorre no local Região da AWS onde os FlexMatch recursos estão localizados. Para reduzir a latência no processo de matchmaking, crie os recursos geograficamente próximos aos jogadores que os usarão. Uma configuração de matchmaking e o conjunto de regras que ela usa devem estar localizados na mesma Região da AWS. Você pode criar FlexMatch recursos em qualquer um dos Regiões da AWS que os suportem.

Para obter mais informações sobre como configurar FlexMatch sua solução de hospedagem, consulte o [guia do Amazon GameLift ServersFlexMatch desenvolvedor](#).

Amazon GameLift Serversna China

Ao usar Amazon GameLift Servers recursos na região da China (Pequim), operada pela Sinnet, ou na região da China (Ningxia), operada pela NWCD, você deve ter uma conta separada (AWS China). Esteja ciente de que alguns recursos não estão disponíveis nas regiões da China. Para obter mais informações sobre o uso Amazon GameLift Servers nessas regiões, consulte os seguintes recursos:

- [Amazon Web Services na China](#)
- [Amazon GameLift Servers](#)(Introdução à Amazon Web Services na China)

Preços para Amazon GameLift Servers

Preços para Amazon GameLift Servers varia de acordo com o tipo de serviço e os recursos que você usa. Para uma discussão completa sobre preços, incluindo exemplos de cenários, consulte [Amazon GameLift Servers Preços](#).

Você pode usar: Amazon GameLift Servers sem incorrer em cobranças com o nível AWS gratuito. Você pode usar os benefícios do nível gratuito se for AWS cliente há menos de 12 meses e estiver dentro dos limites de uso do nível gratuito. Para obter mais informações, consulte [Nível gratuito da AWS](#).

As bases de custo para cada Amazon GameLift Servers os serviços são os seguintes:

- Amazon GameLift Servers hospedagem gerenciada — Você paga com base em (1) uso horário da instância para hospedar sessões de jogos e (2) transferência de dados para tráfego entre clientes de jogos e servidores de jogos hospedados. Os custos de uso da instância variam com base no tipo/tamanho da instância Região da AWS, no sistema operacional e no fato de ser uma instância spot ou sob demanda.
- Amazon GameLift Servers FlexMatch autônomo — Você paga com base em (1) o número de solicitações de jogadores para matchmaking que você faz e (2) o tempo de processamento necessário para avaliar as solicitações de partida e propor partidas.
- Amazon GameLift Servers Hospedagem em qualquer lugar — Você paga com base em (1) o número de sessões de jogo que Amazon GameLift Servers coloca em computadores Anywhere e (2) o número de minutos de conexão do processo do servidor.
- Amazon GameLift Servers FleetiQ — Você paga uma taxa por EC2 cada instância em um grupo de servidores de jogos. A cobrança é derivada do preço por hora da instância e é adicional ao custo EC2 de uso. Os custos de uso da instância variam com base no tipo/tamanho da instância Região da AWS, no sistema operacional e no fato de ser uma instância spot ou sob demanda.

Tópicos

- [Generate Amazon GameLift Servers estimativas de preços](#)
- [Gerencie seu Amazon GameLift Servers custos de hospedagem](#)

Generate Amazon GameLift Servers estimativas de preços

Com AWS Calculadora de Preços, você pode [criar uma estimativa de preço para Amazon GameLift Servers](#). Você não precisa de um conhecimento profundo Conta da AWS ou profundo AWS para usar a calculadora.

AWS Calculadora de Preços A calculadora orienta você nas decisões que afetam os custos do serviço para dar uma ideia de quanto Amazon GameLift Servers pode custar para seu projeto de jogo. Se você ainda não tem certeza de como planeja usar Amazon GameLift Servers, em seguida, use os valores padrão para gerar uma estimativa. Ao planejar o uso da produção, a calculadora pode ajudá-lo a testar possíveis cenários e gerar estimativas mais precisas.

Você pode usar AWS Calculadora de Preços para gerar estimativas para o seguinte Amazon GameLift Servers opções de hospedagem:

- [Estimativa Amazon GameLift Servers hospedagem gerenciada](#)

Estimativa Amazon GameLift Servers hospedagem gerenciada

Essa opção fornece uma estimativa de custo para hospedar seus jogos no Amazon GameLift Servers servidores gerenciados, incluindo os custos de uso da instância do servidor e transferência de dados. With Amazon GameLift Servers hospedagem gerenciada, não há custo adicional para FlexMatch matchmaking.

Se você está hospedando ou planeja hospedar servidores de jogos em mais de uma AWS região ou em mais de um tipo de instância, crie uma estimativa para cada região e tipo de instância.

Amazon GameLift Servers instâncias

Esta seção ajuda você a estimar o tipo e o número de recursos de computação necessários para hospedar sessões de jogo para os jogadores. Amazon GameLift Servers usa [instâncias do Amazon Elastic Compute Cloud \(Amazon EC2\)](#) para gerenciar servidores de jogos. Em Amazon GameLift Servers, você implanta uma frota de instâncias com um tipo de instância e sistema operacional específicos. Caso tenha ou planeja ter várias frotas, crie uma estimativa para cada frota.

Para começar, abra o [Configure Amazon GameLift Servers página](#) de AWS Calculadora de Preços. Adicione uma descrição, escolha uma região e escolha Estimativa Amazon GameLift Servers hospedagem (Instância + Transferência de dados para fora). Em Amazon GameLift Servers instâncias, preencha os seguintes campos:

- Pico de jogadores simultâneos (pico de CCU)

Esse é o número máximo de jogadores que podem se conectar aos seus servidores de jogos ao mesmo tempo. Esse campo indica quanta capacidade de hospedagem Amazon GameLift Servers precisa atender ao pico de demanda dos jogadores. Insira o número máximo diário de jogadores que você espera hospedar usando instâncias na região da AWS escolhida.

Por exemplo, se você quiser permitir que 1.000 jogadores se conectem ao seu jogo ao mesmo tempo, mantenha o valor padrão de **1000**.

- CCU média por hora como porcentagem do pico diário de CCU

Esse é o número médio de jogadores simultâneos por hora em um período de 24 horas. Usamos esse valor para estimar a quantidade de capacidade de hospedagem sustentada que Amazon GameLift Servers precisa ser mantido para seus jogadores. Caso não tenha certeza de qual valor percentual usar, mantenha o valor padrão de **50** por cento. Para jogos com demanda estável de jogadores, é recomendável inserir um valor **70** por cento.

Por exemplo, se seu jogo tiver uma CCU média horária de 6.000 e uma CCU máxima de 10.000, insira o valor de **60** por cento.

- Sessões de jogo por instância

Esse é o número de sessões de jogo que cada uma das instâncias do seu servidor de jogos pode hospedar simultaneamente. Entre os fatores que podem afetar esse número estão as necessidades de recursos do servidor de jogos, o número de jogadores hospedados em cada sessão do jogo e as expectativas de desempenho do jogador. Caso saiba o número de sessões de jogo simultâneas do seu jogo, insira esse valor. Como alternativa, mantenha o valor padrão **20**.

- Jogadores por sessão de jogo

Esse é o número médio de jogadores que se conectam a uma sessão de jogo, conforme definido no design do jogo. Caso tenha modos de jogo com um número diferente de jogadores, estime o número médio de jogadores por sessão de jogo em todo o jogo. O valor padrão é **8**.

- % do buffer de inatividade da instância

Essa é a porcentagem da capacidade de hospedagem não utilizada a ser mantida em reserva para lidar com picos repentinos na demanda dos jogadores. O tamanho do buffer é uma porcentagem do número total de instâncias em uma Frota. O valor padrão **10** por cento.

Por exemplo, com um buffer ocioso de 20%, uma frota que suporta jogadores com 100 instâncias ativas mantém 20 instâncias ociosas.

- Instância spot %

Amazon GameLift Servers as frotas podem usar uma combinação de instâncias sob demanda e instâncias spot. Enquanto as instâncias sob demanda oferecem uma disponibilidade mais confiável, as instâncias spot oferecem uma alternativa altamente econômica. É recomendável usar uma combinação para otimizar a economia e a disponibilidade. Para obter informações sobre como Amazon GameLift Servers usa instâncias spot, consulte [Instâncias sob demanda versus instâncias spot](#).

Nesse campo, insira a porcentagem de instâncias spot a serem mantidas em uma frota. É recomendável uma porcentagem de instância spot entre 50 e 85 por cento. O valor padrão **50** por cento.

Por exemplo, se você implantar uma frota com 100 instâncias e especificar a **40** porcentagem, Amazon GameLift Servers trabalha para manter 60 instâncias sob demanda e 40 instâncias spot.

- Tipo de instância

Amazon GameLift Servers as frotas podem usar uma variedade de tipos de EC2 instâncias da Amazon que variam em capacidade de computação, memória, armazenamento e recursos de rede. Quando você configura um Amazon GameLift Servers frota, escolha o tipo de instância que melhor atenda às necessidades do seu jogo. Para obter informações sobre como selecionar um tipo de instância com Amazon GameLift Servers, consulte [Escolher recursos de computação para uma frota gerenciada](#).

Se você conhece o tipo de instância que está usando ou planeja usar no seu Amazon GameLift Servers frota, escolha esse tipo. Caso não tenha certeza de qual tipo escolher, considere escolher c5.large. Esse é um tipo de alta disponibilidade com tamanho e recursos médios.

- Sistema operacional

Esse campo especifica o sistema operacional no qual seus servidores de jogos são executados no Linux ou Windows. O valor padrão é Linux.

DTO (transferência de dados de saída)

Esta seção ajuda a fazer a estimativa do custo do tráfego entre seus clientes de jogos e servidores de jogos. As taxas de transferência de dados se aplicam somente ao tráfego de saída. A transferência de dados de entrada não tem custo.

No [Configure Amazon GameLift Servers página](#) de AWS Calculadora de Preços, expanda Transferência de dados para fora (DTO) e, em seguida, preencha os seguintes campos:

- Tipo de estimativa de DTO

Escolha estimar o DTO de uma das duas maneiras a seguir, dependendo de como você acompanha a transferência de dados do seu jogo.

- Por mês (em GB) – Caso monitore o tráfego mensal dos seus servidores de jogos, escolha esse tipo.
- Por jogador – Caso acompanhe a transferência de dados por jogador, escolha esse tipo. Esse é o tipo padrão.

No campo a seguir, você estima o DTO por jogador com base no número de horas de jogador calculado na seção anterior.

- DTO por mês (em GB)

Se você escolheu o tipo de estimativa de DTO por mês (em GB), insira seu uso mensal estimado de DTO em GB de cada instância, por região.

- DTO por jogador

Se você escolheu o tipo de estimativa de DTO por jogador, insira o uso estimado de DTO do seu jogo por jogador em KB/s. O valor padrão é **4**.

Quando você terminar de configurar seu Amazon GameLift Servers estimativa de preço, escolha Adicionar à minha estimativa. Para obter mais informações sobre como criar e gerenciar estimativas em AWS Calculadora de Preços, consulte [Criar uma estimativa, configurar um serviço e adicionar mais serviços](#) no Guia do AWS Calculadora de Preços usuário.

Gerencie seu Amazon GameLift Servers custos de hospedagem

Sua AWS fatura reflete seus custos de hospedagem de jogos. Você pode ver as cobranças estimadas do mês atual e as cobranças finais dos meses anteriores no console de faturamento em <https://console.aws.amazon.com/costmanagement/>. Para obter mais informações sobre ferramentas

e recursos para ajudá-lo a gerenciar seus AWS custos, consulte o [Guia AWS Billing do usuário](#). Este guia pode ajudá-lo a analisar o consumo de recursos, estabelecer o uso futuro e determinar suas necessidades de escalabilidade.

Considere estas dicas para ajudar você a gerenciar o custo de Amazon GameLift Servers serviços.

Crie alertas de cobrança para monitorar o uso

Configure um alerta de uso do nível AWS gratuito para notificá-lo quando seu uso estiver próximo ou excedendo os limites do nível gratuito para Amazon GameLift Servers e outros Serviços da AWS. Configure os alertas para agir com base nos seus níveis de uso. Por exemplo, defina automaticamente seu orçamento como zero quando atingir o limite do nível gratuito.

Você também pode definir alertas de CloudWatch cobrança da Amazon para receber notificações quando o uso atingir limites personalizados.

Para obter mais informações, consulte um destes tópicos no Guia do usuário do AWS Billing :

- [Rastreando seu uso do nível AWS gratuito](#)
- [Preferências de alerta de cobrança](#)

Rastreie os custos por Amazon GameLift Servers frota

Use tags de alocação de AWS custos para organizar e monitorar seus custos de hospedagem de jogos com base em Amazon GameLift Servers EC2 Frotas da Amazon e outros recursos. Ao marcar suas frotas, individualmente ou por grupos, crie relatórios de alocação de custos que categorizam os custos com base na etiqueta atribuída. Use esse tipo de relatório para identificar como as frotas estão contribuindo para os custos de hospedagem. Também é possível usar tags para filtrar visualizações no AWS Cost Explorer.

Para obter mais informações, consulte estes tópicos:

- [Usando etiquetas AWS de alocação de custos](#), Guia AWS Billing do usuário
- [Analisando seus AWS custos com o Cost Explorer](#), Guia AWS Cost Management do Usuário

Definir a capacidade da frota gerenciada como zero

As frotas podem continuar a gerar custos, mesmo quando não estão sendo usadas para hospedar sessões de jogos. Para evitar cobranças desnecessárias, [reduza sua frota](#) a zero quando não estiver

em uso. Caso use o ajuste de escala automático, suspenda essa atividade e defina manualmente a capacidade da frota.

Conceitos básicos do Amazon GameLift Servers

Aproveite esses recursos de introdução para saber mais sobre o Amazon GameLift Servers serviço e como você pode começar a desenvolver uma solução de hospedagem personalizada para seus jogos multijogador baseados em sessões.

Antes de começar

- Crie um Conta da AWS (ou designe um existente) para usar com Amazon GameLift Servers.
- Configure usuários com permissões Amazon GameLift Servers e AWS serviços relacionados.
- Selecione um Região da AWS para trabalhar. Para desenvolvimento, escolha uma região próxima à sua localização. Você pode alterar as regiões a qualquer momento.

[Configurar um Conta da AWS](#)

Opções de integração rápida

Experimente essas ferramentas de início rápido para colocar uma solução básica de hospedagem em funcionamento rapidamente com desenvolvimento simplificado. Essas ferramentas são ideais para prova de conceito e prototipagem, ou para usá-las para criar ambientes de teste para desenvolvimento rápido e iterativo de jogos. Depois de usar essas ferramentas para implantar um servidor de jogos para hospedagem, você pode usar o Amazon GameLift Servers console e as ferramentas de API para monitorar o desempenho da frota, gerenciar sessões de jogos e analisar métricas.

- [Embalagem de servidor de jogos para Amazon GameLift Servers](#) — Essa ferramenta e o [tutorial de integração](#) são as maneiras mais rápidas e fáceis de hospedar seu servidor de jogos e executar sessões de jogo Amazon GameLift Servers sem a necessidade de alterações no código. O wrapper Readme do servidor de jogos fornece instruções de linha de comando para todos os tipos de frotas e o tutorial destaca o uso do console e de um tipo de EC2 frota gerenciada para que você possa começar imediatamente. Siga as instruções para se preparar para o gerenciamento de sessões de jogos e a implantação simplificada do servidor de jogos.

Quando você estiver pronto para criar uma solução personalizada de hospedagem de jogos, mude para uma das opções de desenvolvimento personalizadas com integração total com o SDK do servidor para Amazon GameLift Servers. Se seu jogo não precisar de uma solução de hospedagem

personalizada, você poderá continuar usando o invólucro do servidor de jogos para implantar e hospedar servidores de jogos em produção.

- [Amazon GameLift Servers plug-in para Unreal Engine ou Unity](#) — Os plug-ins fornecem fluxos de trabalho de GUI e exemplos de ativos para guiá-lo nas etapas iniciais e implantar seu servidor de jogos com uma solução básica de hospedagem. Use o plug-in para configurar a hospedagem com frotas autogerenciadas do Anywhere ou implantar frotas gerenciadas ou EC2 frotas de contêineres baseadas em nuvem. Quando estiver pronto para desenvolver uma solução de hospedagem personalizada, você poderá desenvolver suas soluções integradas em plug-ins.
- [Kit inicial para contêineres Amazon GameLift Servers gerenciados](#) — Esse kit simplifica as tarefas para integrar um servidor de jogos, preparar uma imagem de contêiner de servidor de jogos e implantar uma frota de contêineres para hospedagem. Para integração, o kit adiciona recursos essenciais de gerenciamento de sessões de jogo ao seu servidor de jogos. O kit usa modelos pré-configurados para criar uma frota de contêineres e um pipeline de implantação automatizado para o servidor do jogo. Quando estiver pronto para adicionar recursos completos de gerenciamento de sessões de jogos, siga um dos roteiros de desenvolvimento personalizados para integrar o SDK do servidor. Amazon GameLift Servers

Opções de desenvolvimento personalizadas

Siga um desses roteiros de desenvolvimento para começar a criar uma solução de hospedagem personalizada completa para seu jogo. Os roteiros fornecem orientações detalhadas sobre como criar, testar e personalizar cada componente em sua solução de hospedagem.

- [Roteiro de desenvolvimento para hospedagem com gerenciamento Amazon GameLift Servers EC2](#)
- [Roteiro de desenvolvimento para hospedagem com contêineres Amazon GameLift Servers gerenciados](#)
- [Roteiro de desenvolvimento para hospedagem com Anywhere Amazon GameLift Servers](#)
- [Roteiro de desenvolvimento para hospedagem híbrida com Amazon GameLift Servers](#)

Exemplos do Amazon GameLift Servers

Se você está pensando em usar Amazon GameLift Servers para gerenciar seu servidor de jogo personalizado ou está interessado em tirar proveito dele Amazon GameLift Servers Realtime, recomendamos que você experimente os exemplos a seguir antes de usar o serviço para seu próprio jogo. O exemplo de servidor de jogos personalizado oferece experiência com hospedagem de jogos

no Amazon GameLift Servers console. O Amazon GameLift Servers Realtime exemplo mostra como preparar um jogo para hospedagem usando Realtime servidores.

Exemplo de servidor de jogo personalizado

Este exemplo demonstra o processo de implantação de um servidor de jogos de amostra em uma EC2 frota Amazon GameLift Servers gerenciada para hospedagem. Use o exemplo de cliente de jogo para se conectar a uma sessão de jogo ao vivo. Você pode experimentar como usar Amazon GameLift Servers .tools, incluindo o console e a AWS CLI, para monitorar o desempenho e o uso da hospedagem da frota.

O exemplo conduz você pelas etapas a seguir:

- Faça upload da compilação do servidor de jogos de demonstração.
- Crie uma frota para executar a compilação do servidor de jogos.
- Obtenha o exemplo de cliente de jogo e use-o para se conectar a um servidor de jogos e participar de uma sessão de jogo.
- Analise as métricas da frota e da sessão de jogo.

Execute vários clientes de jogos e jogue para gerar dados de hospedagem. Use o Amazon GameLift Servers console para visualizar recursos de hospedagem, monitorar métricas e explorar opções para escalar a capacidade de hospedagem da frota.

Para começar, entre no [Amazon GameLift Serversconsole](#). Na navegação do lado esquerdo, acesse Recursos, Testar um jogo de demonstração.

Amazon GameLift ServersRealtimeexemplo

Este exemplo é um tutorial completo que mostra como implantar um exemplo de jogo multijogador, Mega Frog Race, com Amazon GameLift ServersRealtime. O tutorial aborda como integrar seu cliente de jogo ao Realtime SDK e implantar uma solução de hospedagem completa com Realtime servidores em EC2 frotas gerenciadas.

Para ver um tutorial prático, consulte [Criação de servidores para jogos móveis multijogador com apenas algumas linhas JavaScript](#) no blog AWS for Games. Para o código-fonte do Mega Frog Race, consulte o [GitHub repositório](#).

O código-fonte inclui as seguintes partes:

- Cliente de jogo — Código-fonte do cliente de jogo C++, criado no Unity. O cliente do jogo obtém informações de conexão da sessão do jogo, se conecta ao servidor e troca atualizações com outros jogadores.
- Serviço de back-end — código-fonte de uma AWS Lambda função que gerencia chamadas diretas para a API do serviço para Amazon GameLift Servers.
- Realtiescript — Um arquivo de script de origem que configura uma frota de Realtime servidores para o jogo. Esse script inclui a configuração mínima necessária para que cada Realtime servidor se comunique Amazon GameLift Servers e hospede sessões de jogo.

Depois de configurar o jogo de amostra para hospedagem, use-o como ponto de partida para experimentar outros Amazon GameLift Servers recursos, como FlexMatch.

Configurar um Conta da AWS

Para começar a usar Amazon GameLift Servers, crie e configure seu Conta da AWS. Não há custo para criar uma Conta da AWS. Esta seção explica como criar sua conta, configurar seus usuários e configurar permissões.

Tópicos

- [Inscreva-se para um Conta da AWS](#)
- [Criar um usuário com acesso administrativo](#)
- [Definir permissões de usuário para Amazon GameLift Servers](#)
- [Configurar o acesso programático, consulte para usuários](#)
- [Configurar o acesso programático para o jogo](#)
- [Exemplos de permissão do IAM para Amazon GameLift Servers](#)
- [Configure uma função de serviço do IAM para Amazon GameLift Servers](#)

Inscreva-se para um Conta da AWS

Se você não tiver um Conta da AWS, conclua as etapas a seguir para criar um.

Para se inscrever em um Conta da AWS

1. Abra a <https://portal.aws.amazon.com/billing/inscrição>.

2. Siga as instruções online.

Parte do procedimento de inscrição envolve receber uma chamada telefônica ou uma mensagem de texto e inserir um código de verificação pelo teclado do telefone.

Quando você se inscreve em uma Conta da AWS, um Usuário raiz da conta da AWS é criado. O usuário-raiz tem acesso a todos os Serviços da AWS e recursos na conta. Como prática recomendada de segurança, atribua o acesso administrativo a um usuário e use somente o usuário-raiz para executar [tarefas que exigem acesso de usuário-raiz](#).

AWS envia um e-mail de confirmação após a conclusão do processo de inscrição. A qualquer momento, você pode visualizar a atividade atual da sua conta e gerenciar sua conta acessando <https://aws.amazon.com/e> escolhendo Minha conta.

Criar um usuário com acesso administrativo

Depois de se inscrever em uma Conta da AWS, proteja seu Usuário raiz da conta da AWS AWS IAM Identity Center, habilite e crie um usuário administrativo para que você não use o usuário root nas tarefas diárias.

Proteja seu Usuário raiz da conta da AWS

1. Faça login [AWS Management Console](#) como proprietário da conta escolhendo Usuário raiz e inserindo seu endereço de Conta da AWS e-mail. Na próxima página, insira a senha.

Para obter ajuda ao fazer login usando o usuário-raiz, consulte [Fazer login como usuário-raiz](#) no Guia do usuário do Início de Sessão da AWS .

2. Habilite a autenticação multifator (MFA) para o usuário-raiz.

Para obter instruções, consulte [Habilitar um dispositivo de MFA virtual para seu usuário Conta da AWS raiz \(console\) no Guia](#) do usuário do IAM.

Criar um usuário com acesso administrativo

1. Habilita o Centro de Identidade do IAM.

Para obter instruções, consulte [Habilitar o AWS IAM Identity Center](#) no Guia do usuário do AWS IAM Identity Center .

2. No Centro de Identidade do IAM, conceda o acesso administrativo a um usuário.

Para ver um tutorial sobre como usar o Diretório do Centro de Identidade do IAM como fonte de identidade, consulte [Configurar o acesso do usuário com o padrão Diretório do Centro de Identidade do IAM](#) no Guia AWS IAM Identity Center do usuário.

Iniciar sessão como o usuário com acesso administrativo

- Para fazer login com o seu usuário do Centro de Identidade do IAM, use o URL de login enviado ao seu endereço de e-mail quando o usuário do Centro de Identidade do IAM foi criado.

Para obter ajuda para fazer login usando um usuário do IAM Identity Center, consulte [Como fazer login no portal de AWS acesso](#) no Guia Início de Sessão da AWS do usuário.

Atribuir acesso a usuários adicionais

1. No Centro de Identidade do IAM, crie um conjunto de permissões que siga as práticas recomendadas de aplicação de permissões com privilégio mínimo.

Para obter instruções, consulte [Criar um conjunto de permissões](#) no Guia do usuário do AWS IAM Identity Center .

2. Atribua usuários a um grupo e, em seguida, atribua o acesso de autenticação única ao grupo.

Para obter instruções, consulte [Adicionar grupos](#) no Guia do usuário do AWS IAM Identity Center .

Definir permissões de usuário para Amazon GameLift Servers

Crie usuários adicionais ou estenda as permissões de acesso aos usuários existentes conforme necessário para seus Amazon GameLift Servers recursos. Como prática recomendada ([Melhores práticas de segurança no IAM](#)), aplique permissões de privilégio mínimo para todos os usuários. Para obter orientação sobre a sintaxe de permissões, consulte [Exemplos de permissão do IAM para Amazon GameLift Servers](#).

Use as instruções a seguir para definir permissões de usuário com base em como você gerencia os usuários em sua AWS conta.

Para conceder acesso, adicione as permissões aos seus usuários, grupos ou perfis:

- Usuários e grupos em AWS IAM Identity Center:

Crie um conjunto de permissões. Siga as instruções em [Criação de um conjunto de permissões](#) no Guia do usuário do AWS IAM Identity Center .

- Usuários gerenciados no IAM com provedor de identidades:

Crie um perfil para a federação de identidades. Siga as instruções em [Criando um perfil para um provedor de identidades de terceiros \(federação\)](#) no Guia do Usuário do IAM.

- Usuários do IAM:
 - Crie um perfil que seu usuário possa assumir. Siga as instruções em [Criação de um perfil para um usuário do IAM](#) no Guia do usuário do IAM.
 - (Não recomendado) Vincule uma política diretamente a um usuário ou adicione um usuário a um grupo de usuários. Siga as instruções em [Adição de permissões a um usuário \(console\)](#) no Guia do usuário do IAM.

Ao trabalhar com usuários do IAM, como melhor prática, sempre atribua permissões a funções ou grupos de usuários e não a usuários individuais.

Configurar o acesso programático, consulte para usuários

Os usuários precisam de acesso programático se quiserem interagir com pessoas AWS fora do AWS Management Console. A forma de conceder acesso programático depende do tipo de usuário que está acessando AWS.

Para conceder acesso programático aos usuários, selecione uma das seguintes opções:

Qual usuário precisa de acesso programático?	Para	Por
Identidade da força de trabalho (Usuários gerenciados no Centro de Identidade do IAM)	Use credenciais temporárias para assinar solicitações programáticas para o AWS CLI AWS SDKs, ou. AWS APIs	Siga as instruções da interface que deseja utilizar. • Para o AWS CLI, consulte Configurando o AWS CLI para uso AWS IAM Identity Center no Guia do AWS Command Line Interface usuário.

Qual usuário precisa de acesso programático?	Para	Por
		Para AWS SDKs, ferramentas e AWS APIs, consulte a autenticação do IAM Identity Center no Guia de referência de ferramentas AWS SDKs e ferramentas.
IAM	Use credenciais temporárias para assinar solicitações programáticas para o AWS CLI AWS SDKs, ou. AWS APIs	Siga as instruções em Como usar credenciais temporárias com AWS recursos no Guia do usuário do IAM.
IAM	(Não recomendado) Use credenciais de longo prazo para assinar solicitações programáticas para o AWS CLI, AWS SDKs, ou. AWS APIs	Siga as instruções da interface que deseja utilizar. - Para isso AWS CLI, consulte Autenticação usando credenciais de usuário do IAM no Guia do AWS Command Line Interface usuário. - Para ferramentas AWS SDKs e ferramentas, consulte Autenticar usando credenciais de longo prazo no Guia de referência de ferramentas AWS SDKs e ferramentas. - Para isso AWS APIs, consulte Gerenciamento de chaves de acesso para usuários do IAM no Guia do usuário do IAM.

Se você usa chaves de acesso, consulte [Práticas recomendadas para gerenciar chaves de AWS acesso](#).

Configurar o acesso programático para o jogo

A maioria dos jogos usa serviços de back-end para se comunicar Amazon GameLift Servers usando o AWS SDKs. Use um serviço de backend (representando um cliente de jogo) para solicitar sessões de jogo, posicionar jogadores em partidas e realizar outras tarefas. Esses serviços precisam de acesso programático e credenciais de segurança para autenticar chamadas para a API do serviço Amazon GameLift Servers.

Para Amazon GameLift Servers, você gerencia esse acesso criando um usuário player no AWS Identity and Access Management (IAM). Gerencie as permissões do usuário jogador por meio de uma das seguintes opções:

- Crie um perfil do IAM com permissões de usuário jogador e permita que o usuário jogador assuma o perfil quando necessário. O serviço de back-end deve incluir código para assumir essa função antes de fazer solicitações para Amazon GameLift Servers. De acordo com as melhores práticas de segurança, os perfis fornecem acesso limitado e temporário. Você pode usar funções para cargas de trabalho executadas em AWS recursos ([funções do IAM](#)) ou fora deles AWS ([funções do IAM em qualquer lugar](#)).
- Crie um grupo de usuários do IAM com permissões de usuário do jogador e adicione seu usuário jogador ao grupo. Essa opção fornece credenciais de usuário de longo prazo ao jogador, que o serviço de back-end deve armazenar e usar ao se comunicar. Amazon GameLift Servers

Para obter a sintaxe da política de permissões, consulte [Exemplos de permissão de usuário do jogador](#).

Para obter mais informações sobre o gerenciamento de permissões para uso por um workload, consulte [Identidades do IAM: credenciais temporárias no IAM](#).

Exemplos de permissão do IAM para Amazon GameLift Servers

Use a sintaxe nesses exemplos para definir permissões AWS Identity and Access Management (IAM) para usuários que precisam acessar Amazon GameLift Servers recursos. Para obter mais informações sobre como gerenciar permissões de usuário, consulte [Definir permissões de usuário para Amazon GameLift Servers](#). Ao gerenciar permissões para usuários fora do IAM Identity Center,

como melhor prática, sempre atribua permissões ao perfil do IAM ou grupos de usuários, não a usuários individuais.

Se você estiver usando Amazon GameLift Servers FleetIQ como uma solução independente, consulte [Configurar seu Conta da AWS para Amazon GameLift ServersFleetIQ](#).

Exemplos de permissões administrativas

Esses exemplos fornecem a um administrador ou desenvolvedor de hospedagem acesso direcionado para gerenciar recursos de hospedagem de Amazon GameLift Servers jogos.

Example Sintaxe para permissões de recursos de acesso Amazon GameLift Servers total

O exemplo a seguir estende o acesso total a todos os Amazon GameLift Servers recursos.

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

Example Sintaxe para permissões Amazon GameLift Servers de recursos com suporte para regiões que não estão habilitadas por padrão

O exemplo a seguir estende o acesso a todos os Amazon GameLift Servers recursos e AWS regiões que não estão habilitados por padrão. Para obter mais informações sobre regiões que não estão habilitadas por padrão e como habilitá-las, consulte [Gerenciando Regiões da AWS](#) no Referência geral da AWS.

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

```
}  
}
```

Example Sintaxe do Amazon GameLift Servers recurso para acessar imagens de contêineres no Amazon ECR

O exemplo a seguir estende o acesso às ações do Amazon Elastic Container Registry (Amazon ECR) Amazon GameLift Servers que os usuários precisam ao trabalhar com frotas de contêineres gerenciadas.

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

Example Sintaxe para Amazon GameLift Servers recursos e permissões **PassRole**

O exemplo a seguir amplia o acesso a todos os Amazon GameLift Servers recursos e permite que um usuário transmita uma função de serviço do IAM para Amazon GameLift Servers o. Uma função de serviço oferece capacidade Amazon GameLift Servers limitada de acessar outros recursos e serviços em seu nome, conforme descrito em [Configure uma função de serviço do IAM para Amazon GameLift Servers](#). Por exemplo, ao responder a uma CreateBuild solicitação, Amazon GameLift Servers precisa acessar seus arquivos de compilação em um bucket do Amazon S3. Para obter mais informações sobre a PassRole ação, consulte [IAM: passar uma função do IAM para um AWS serviço específico](#) no Guia do usuário do IAM.

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



```
    },
    {
      "Effect": "Allow",
      "Action": "iam:PassRole",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "iam:PassedToService": "gamelift.amazonaws.com"
        }
      }
    }
  ]
}
```

Exemplos de permissão de usuário do jogador

Esses exemplos permitem que um serviço de back-end ou outra entidade faça chamadas de API para a Amazon GameLift Servers API. Eles abrangem os cenários comuns para gerenciar sessões de jogo, sessões de jogadores e marcação de jogos. Consulte mais detalhes em [Configurar o acesso programático para o jogo](#).

Exemplo Sintaxe das permissões de colocação de sessão de jogo

O exemplo a seguir estende o acesso às Amazon GameLift Servers APIs que usam filas de posicionamento de sessões de jogo para criar sessões de jogo e gerenciar sessões de jogadores.

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

Example Sintaxe para permissões de marcação de jogos

O exemplo a seguir estende o acesso aos Amazon GameLift Servers APIs que gerenciam as atividades de FlexMatch matchmaking. FlexMatch combina jogadores para sessões de jogo novas ou existentes e inicia a colocação da sessão de jogo para jogos hospedados em Amazon GameLift Servers. Para obter mais informações sobre FlexMatch, consulte [O que é Amazon GameLift Servers FlexMatch?](#)

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

Example Sintaxe para permissões manuais de posicionamento de sessões de jogos

O exemplo a seguir estende o acesso aos Amazon GameLift Servers APIs que criam manualmente sessões de jogo e sessões de jogadores em frotas especificadas. Esse cenário oferece suporte a jogos que não usam filas de posicionamento, como jogos que permitem que os jogadores participem escolhendo em uma lista de sessões de jogo disponíveis (o método "list-and-pick").

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "PlayerPermissionsForManualGameSessions",
    "Effect": "Allow",
    "Action": [
      "gamelift:CreateGameSession",
      "gamelift:DescribeGameSessions",
      "gamelift:SearchGameSessions",
      "gamelift:CreatePlayerSession",

```

```
    "gamelift:CreatePlayerSessions",  
    "gamelift:DescribePlayerSessions"  
  ],  
  "Resource": "*"   
}
```

Configure uma função de serviço do IAM para Amazon GameLift Servers

Alguns Amazon GameLift Servers recursos exigem que você estenda o acesso limitado a outros AWS recursos que você possui. Você pode fazer isso criando uma função AWS Identity and Access Management (IAM). Um [perfil do IAM](#) é uma identidade do IAM que você pode criar em sua conta que tem permissões específicas. Uma função do IAM é semelhante à de um usuário do IAM, pois é uma AWS identidade com políticas de permissões que determinam o que a identidade pode ou não fazer AWS. No entanto, em vez de ser exclusivamente associada a uma pessoa, o propósito do perfil é ser assumido por qualquer pessoa que precisar dele. Além disso, um perfil não tem credenciais de longo prazo padrão associadas a ele, como senha ou chaves de acesso. Em vez disso, quando você assumir um perfil, ele fornecerá credenciais de segurança temporárias para sua sessão de perfil.

Este tópico aborda como criar uma função que você possa usar com suas frotas Amazon GameLift Servers gerenciadas. Se você usa Amazon GameLift Servers FleetIQ para otimizar a hospedagem de jogos em suas instâncias do Amazon Elastic Compute Cloud (Amazon EC2), consulte [Configurar seu Conta da AWS para Amazon GameLift ServersFleetIQ](#).

No procedimento a seguir, crie uma função com uma política de permissões personalizada e uma política de confiança que Amazon GameLift Servers permita assumir a função.

Crie uma função de serviço do IAM para uma EC2 frota Amazon GameLift Servers gerenciada

Etapa 1: criar uma política de permissões.

Use as instruções e os exemplos nesta página para criar uma política de permissões personalizada para o tipo de Amazon GameLift Servers frota com a qual você está trabalhando.

Para usar o editor de políticas JSON para criar uma política

1. Faça login no AWS Management Console e abra o console do IAM em <https://console.aws.amazon.com/iam/>.

2. No painel de navegação à esquerda, escolha Políticas.

Se essa for a primeira vez que você escolhe Políticas, a página Bem-vindo às políticas gerenciadas será exibida. Escolha Começar.

3. Na parte superior da página, escolha Criar política.
4. Na seção Editor de políticas, escolha a opção JSON.
5. Insira ou cole um documento de política JSON. Para obter detalhes sobre a linguagem da política do IAM, consulte a referência de [política JSON do IAM](#).
6. Resolva os avisos de segurança, erros ou avisos gerais gerados durante a [validação de política](#) e depois escolha Próximo.

 Note

Você pode alternar entre as opções de editor Visual e JSON a qualquer momento. Porém, se você fizer alterações ou escolher Próximo no editor Visual, o IAM poderá reestruturar a política a fim de otimizá-la para o editor visual. Para obter mais informações, consulte [Reestruturação de política](#) no Guia do usuário do IAM.

7. (Opcional) Ao criar ou editar uma política no AWS Management Console, você pode gerar um modelo de política JSON ou YAML que pode ser usado em AWS CloudFormation modelos.

Para fazer isso, no editor de políticas, escolha Ações e, em seguida, escolha Gerar CloudFormation modelo. Para saber mais AWS CloudFormation, consulte a [referência do tipo de AWS Identity and Access Management recurso](#) no Guia AWS CloudFormation do usuário.

8. Quando terminar de adicionar as permissões à política, escolha Avançar.
9. Na página Revisar e criar, insira um Nome de política e uma Descrição (opcional) para a política que você está criando. Revise Permissões definidas nessa política para ver as permissões que são concedidas pela política.
10. (Opcional) Adicione metadados à política associando tags como pares de chave-valor. Para obter mais informações sobre o uso de tags no IAM, consulte [Tags para AWS Identity and Access Management recursos](#) no Guia do usuário do IAM.
11. Escolha Criar política para salvar sua nova política.

Etapa 2: Crie uma função que Amazon GameLift Servers possa assumir.

Para criar um perfil do IAM

1. No painel de navegação do console do IAM, escolha Perfis e, em seguida, Criar perfil.
2. Na página Selecionar entidade confiável, escolha a opção Política de confiança personalizada. Esta seleção abre o editor de Política de confiança personalizada.
3. Substitua a sintaxe JSON padrão pela seguinte e, em seguida, escolha Avançar para continuar.

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

4. Na página Adicionar permissões, localize e selecione a política de permissões criada na Etapa 1. Escolha Avançar para continuar.
5. Na página Nome, revisar e criar, insira um Nome de perfil e uma Descrição (opcional) para o perfil que você está criando. Revise as Entidades confiáveis e as Permissões adicionadas.
6. Escolha Criar perfil para salvar seu novo perfil.

Crie uma função do IAM para contêineres Amazon GameLift Servers gerenciados

Se você estiver usando contêineres Amazon GameLift Servers gerenciados, precisará criar uma função de serviço do IAM para uso com uma frota de contêineres. Essa função concede permissões limitadas Amazon GameLift Servers necessárias para gerenciar os recursos da sua frota de contêineres e realizar ações em seu nome.

Para criar uma função do IAM para uma frota de contêineres

1. Faça login no AWS Management Console e abra o console do IAM em <https://console.aws.amazon.com/iam/>.
2. No painel de navegação do console do IAM, escolha Perfis e, em seguida, Criar perfil.

3. Na página Selecionar entidade confiável, escolha AWS serviço e selecione o caso de uso "GameLift". Escolha Próximo.
4. Em Adicionar permissões, escolha a política gerenciada `GameLiftContainerFleetPolicy`. Escolha Próximo. Consulte [AWS políticas gerenciadas para Amazon GameLift Servers](#) para obter mais informações sobre essa política.
5. Em Nome, revisar e criar, insira um nome de função e escolha Criar função para salvar a nova função.

Sintaxe da política de permissões

- Permissões Amazon GameLift Servers para assumir a função de serviço

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

- Permissões para acessar AWS regiões que não estão habilitadas por padrão

```
{
  "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"  
  }  
]  
}
```

Obtenha ferramentas Amazon GameLift Servers de desenvolvimento

Amazon GameLift Servers fornece um conjunto SDKs e outras ferramentas para ajudá-lo a criar soluções de hospedagem de jogos para seus jogos. Eles SDKs adicionam funcionalidades a servidores de jogos, clientes de jogos e serviços de back-end que permitem que eles interajam com o Amazon GameLift Servers serviço. Para obter as informações mais recentes sobre versões e compatibilidade do Amazon GameLift Servers SDK, consulte [Notas de release do Amazon GameLift Servers](#).

Para servidores de jogos

Integre e crie seus servidores de jogos de 64 bits com o SDK do servidor para Amazon GameLift Servers. O servidor do jogo usa o SDK do servidor para se comunicar com o Amazon GameLift Servers serviço para gerenciamento de sessões de jogo, incluindo iniciar, atualizar e interromper sessões de jogo. Para obter ajuda com a integração do SDK do servidor em seus projetos de jogos, consulte [Preparando jogos para Amazon GameLift Servers](#)

Suporte ao desenvolvimento

- Sistema operacional de desenvolvimento
 - Windows
 - Linux
- Linguagens de programação

[Obtenha o Amazon GameLift Servers SDK](#). Para informações específicas de versão e instruções de instalação, consulte os arquivos readme incluídos em cada pacote.

- [SDK do servidor C++](#)
 - [Referência do SDK do servidor](#)
 - [Como integrar](#)

- [SDK do servidor C#](#) (Support para .NET 4, .NET 6, .NET 8 varia de acordo com a versão, consulte) [Versões do SDK](#)
 - [Referência do SDK do servidor](#)
 - [Como integrar](#)
- [SDK do servidor Go](#)
 - [Referência do SDK do servidor](#)
 - [Como integrar](#)
- Suporte ao motor de jogo

O plug-in completo Amazon GameLift Servers inclui fluxos de trabalho de interface do usuário e amostras de ativos, bem como versões integradas do AWS SDK e do SDK do servidor. Os fluxos de trabalho orientam você sobre como configurar e implantar seu servidor de jogos para hospedagem com frotas gerenciadas, EC2 frotas de contêineres gerenciadas ou frotas autogerenciadas em qualquer lugar. Se você não precisar dos fluxos de trabalho guiados, poderá obter apenas o SDK do servidor para seu mecanismo de jogo nos mesmos GitHub repositórios.

Se você estiver usando outro mecanismo de jogo ou ambiente de desenvolvimento não compatível com o plug-in, obtenha o SDK do servidor para sua linguagem de programação e adicione-o ao seu projeto de jogo.

Para informações específicas de versão e instruções de instalação, consulte os arquivos readme incluídos em cada pacote.

- [Plugin para Unreal Engine](#) — Criado para uso com as versões 5.5 do Unreal Engine. Verifique os readmes específicos da versão para obter suporte ao Unreal.
 - [Guia de plug-ins para Unreal Engine](#)
 - [SDK 5.x do servidor C++ \(Unreal\) para -- Ações Amazon GameLift Servers](#)
- [Plugin para Unity](#) — Criado para uso com versões LTS do Unity Editor 6.0, 2022.3 ou 2021.3. Ele suporta os perfis .NET Framework e .NET Standard do Unity, com o .NET Standard 2.1 e .NET 4.x. Verifique os readmes específicos da versão para suporte ao Unity.
 - [Guia de plug-ins para Unity](#)
 - [Referência do SDK do servidor C#](#)
- [SDK de servidor para Unreal](#)
 - [Referência do SDK do servidor](#)
 - [Amazon GameLift ServersIntegre-se a um projeto Unreal Engine](#)

- [SDK de servidor para Unity](#)
- [Referência do SDK do servidor](#)
- [Amazon GameLift ServersIntegre-se a um projeto Unity](#)

Suporte de tempo de execução

Para uma solução de hospedagem gerenciada, crie seu servidor de jogos para ser executado em uma das seguintes imagens de máquina da Amazon (AMIs). Consulte [Amazon GameLift ServersVersões da AMI](#) Amazon GameLift Servers para obter mais detalhes.

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

Note

O Amazon Linux 2 (AL2) chegará ao fim do suporte em 30 de junho de 2025. Veja mais detalhes no [Amazon Linux 2 FAQs](#). Para servidores de jogos hospedados AL2 e que usam o SDK 4.x Amazon GameLift Servers do servidor, primeiro atualize a compilação do servidor de jogos para o SDK 5.x do servidor e, em seguida, implante em 023 instâncias. AL2 Consulte [Migre para o SDK 5.x do servidor para Amazon GameLift Servers](#).

Ferramentas adicionais

[Embalagem de servidor de jogos para Amazon GameLift Servers](#)

Essa ferramenta ajuda você a implantar um servidor de jogos para hospedagem com um conjunto de funcionalidades básicas de gerenciamento de sessão de jogo. Com essa ferramenta, você não precisa fazer alterações no código do jogo ou integrar o SDK do servidor para Amazon GameLift Servers. Use o wrapper do servidor de jogos para empacotar seu servidor de jogos e implantá-lo para hospedagem de jogos usando qualquer uma das três soluções de Amazon GameLift Servers hospedagem (em qualquer lugar, contêineres gerenciados EC2 ou gerenciados). Essa ferramenta é mais adequada para avaliação precoce ou criação de protótipos com seu próprio jogo ou um jogo de amostra, pois não oferece suporte à personalização do servidor de jogos. Se seu jogo não precisar

de recursos personalizados, você pode implantar seu servidor de jogos com o invólucro do servidor de jogos para hospedagem de produção.

[Amazon GameLift ServersKit de ferramentas](#)

O Amazon GameLift Servers kit de ferramentas é uma coleção de scripts e outras ferramentas que desenvolvemos para ajudar os desenvolvedores com cenários e problemas comuns. Os materiais do kit de ferramentas incluem scripts, código de amostra e readmes.

- [Kit inicial de contêineres](#) — Use essa ferramenta para simplificar as tarefas de configuração de compilações de servidores de jogos para hospedagem com Amazon GameLift Servers contêineres gerenciados. O kit integra recursos essenciais de gerenciamento de sessões de jogos em um servidor de jogos e usa modelos pré-configurados para criar uma frota de contêineres e configurar um pipeline de implantação automatizado para a construção do seu servidor de jogos. Após a implantação, você pode monitorar o desempenho da frota, gerenciar sessões de jogo e analisar métricas usando o Amazon GameLift Servers console e as ferramentas de API. O kit se integra à automação AWS CodeBuild de construção, ao Amazon Simple Storage Service para armazenamento e AWS CloudFormation à implantação de infraestrutura.
- [Ferramenta de atualização rápida de compilação](#) — Use essa ferramenta para modificar uma compilação de servidor de jogo que já está implantada em uma EC2 frota gerenciada. A ferramenta foi criada para ajudá-lo a trocar rapidamente os arquivos de compilação do jogo sem precisar configurar e criar novas EC2 frotas a cada alteração. Você pode atualizar instâncias individuais ou todas as instâncias da frota. As opções permitem que você substitua arquivos de compilação específicos ou uma compilação inteira e gerencie como reiniciar os servidores de jogos após as atualizações.

Para serviços de clientes de jogos

Crie um serviço de back-end de 64 bits para seus clientes de jogos usando o AWS SDK, que inclui a API de serviço para Amazon GameLift Servers. O serviço de back-end do seu jogo lida com as interações do lado do cliente com o Amazon GameLift Servers serviço, incluindo iniciar novas sessões de jogo e juntar jogadores aos jogos.

[Obtenha o AWS SDK](#)

Para obter mais informações sobre como usar o AWS SDK com Amazon GameLift Servers, consulte os seguintes recursos:

- [Referência de API do Amazon GameLift Servers](#)

- [Integração do serviço do cliente](#)
- [Projete um serviço de back-end para o cliente](#)

Para gerenciamento Amazon GameLift Servers de recursos

Use as ferramentas a seguir para criar, atualizar e monitorar seus recursos de hospedagem Amazon GameLift Servers gerenciada.

- [AWS Management Console](#)— O AWS console é um aplicativo baseado na web que fornece acesso centralizado a todos os consoles de AWS serviço individuais, inclusive. Amazon GameLift Servers Use o console para criar ou entrar em uma AWS conta e abra o Amazon GameLift Servers console para trabalhar com seus recursos de hospedagem de jogos. Você pode configurar e implantar frotas de hospedagem e outros recursos, visualizar métricas de uso e desempenho, rastrear recursos no painel e muitas outras tarefas. [Vá para o Amazon GameLift Servers console.](#)
- [API de serviço para Amazon GameLift Servers](#) — Essa API oferece acesso programático a todos os seus Amazon GameLift Servers recursos. Ele faz parte do AWS SDK, que você pode baixar para uso com as linguagens de programação mais populares. [Obtenha o AWS SDK.](#)
- [AWS interface de linha de comando \(CLI\)](#) — A AWS CLI permite que você interaja com AWS serviços usando um shell de linha de comando. As ferramentas fornecem acesso direto ao público APIs para AWS serviços, bem como comandos personalizados que estão disponíveis para um serviço. [Obtenha a AWS CLI.](#)
- [AWS CloudFormation](#) para Amazon GameLift Servers — O AWS CloudFormation serviço ajuda você a modelar e configurar AWS recursos para simplificar a implantação e o gerenciamento da infraestrutura. Crie um AWS CloudFormation modelo para descrever os Amazon GameLift Servers recursos da sua solução de hospedagem e, em seguida, use o modelo para criar recursos adicionais ou atualizar configurações. Veja a [referência do tipo de Amazon GameLift Servers recurso.](#)

Para Amazon GameLift ServersRealtime

Configure e implante Realtime servidores para hospedar seus jogos multijogador. Para permitir que seus clientes do jogo se conectem aos Realtime servidores, use o SDK Amazon GameLift Servers Realtime do cliente. Para começar, [baixe o SDK Realtime do cliente](#). Para obter informações sobre a configuração, consulte [Integrando um cliente de jogo para Amazon GameLift ServersRealtime](#).

Compatibilidade com o SDK

O SDK do cliente Realtime contém origem para as seguintes linguagens:

- C# (.NET)

Ambientes de desenvolvimento

Compile o SDK a partir da origem conforme necessário para os seguintes sistemas operacionais de desenvolvimento e mecanismos de jogos compatíveis:

- Sistemas operacionais – Windows, Linux, Android, iOS.
- Mecanismos de jogos – Unity, mecanismos que oferecem suporte para bibliotecas C#

Sistemas operacionais do servidor de jogos

Você pode implantar Realtime servidores em recursos de hospedagem que são executados nas seguintes plataformas:

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

Note

AL2 está chegando ao fim do suporte. Veja mais detalhes no [Amazon Linux 2 FAQs](#).

Tutorial: Integração rápida com o Amazon GameLift Servers envólucro

Bem-vindo ao tutorial de integração do Amazon GameLift Servers. Neste tutorial, você implanta rapidamente seu servidor de jogos para ser hospedado em uma frota de recursos computacionais baseados em nuvem. Use este tutorial para evitar o trabalho de integrar o SDK do servidor ao código do jogo e, em vez disso, implante seu jogo com a funcionalidade mínima para se comunicar com o Amazon GameLift Servers serviço e executar sessões de jogo. Você configurará uma solução básica de hospedagem e a usará para experimentar toda a gama de recursos, como escalonamento automático e matchmaking. Também é uma ótima maneira de hospedar um protótipo do seu jogo como parte de uma demonstração ao vivo ou para testes.

Principais benefícios desse método de integração:

- Implante rapidamente seu servidor de jogos para hospedagem rápida.
- Nenhuma alteração no código do jogo e nenhuma modificação necessária.
- Use esse método com qualquer executável de jogo, independentemente do motor de jogo.
- Explore todas as ferramentas Amazon GameLift Servers de gerenciamento, incluindo o monitoramento da atividade da sessão de jogo e a integridade da hospedagem.

Note

O invólucro é destinado à avaliação e ao uso básico na produção. Recursos avançados, como gerenciamento detalhado da sessão do jogador, exigem uma integração completa do SDK do servidor.

Pré-requisitos

Antes de começar, verifique se você tem:

- E Conta da AWS com as permissões apropriadas
- AWS CLI instalado
- Vá 1.18+
- Um servidor de jogos multijogador executável
- Faça (Linux/Mac)
- Git instalado com uma conta ativa

Visão geral

Neste tutorial, você irá:

1. Obtenha e construa o invólucro
2. Prepare a versão do jogo
3. Configurar o invólucro
4. Faça o upload da versão do servidor do jogo
5. Crie a EC2 frota gerenciada

6. Crie e conecte-se a uma sessão de jogo
7. Monitore e gerencie seus servidores de jogos
8. Dimensione seus servidores de jogos

Etapa 1: Obtenha e construa o invólucro do servidor de jogos

Use os comandos a seguir para obter a fonte do wrapper do servidor de jogos e criar o wrapper. Esses comandos usam SSH, mas você também pode acessar diretamente o repositório do 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
```

Em uma compilação bem-sucedida, um diretório “de saída” é adicionado ao `amazon-gamelift-servers-game-server-wrapper`. Nesse diretório, há três pastas, uma para cada opção de frota de hospedagem compatível, que contêm um conjunto de artefatos de construção. Neste tutorial, você está implantando em EC2 frotas gerenciadas, então você usará a pasta `gamelift-servers-managed-ec2`.

Etapa 2: Prepare a versão do seu servidor de jogo

Nesta etapa, você prepara os arquivos de compilação do servidor de jogos para serem enviados para o Amazon GameLift Servers.

Crie o diretório do jogo

Agora, prepare um diretório de jogo em sua máquina local. Esse diretório deve conter todos os arquivos que você precisa para executar seu servidor de jogos Amazon GameLift Servers. Isso inclui o wrapper do servidor do jogo, a compilação do servidor do jogo e o `config.yaml` que faz o wrapper funcionar com o servidor do jogo.

Use as seguintes etapas:

1. EC2 Frota gerenciada. Na pasta de saída do wrapper do servidor do jogo, encontre o artefato de construção para implantação em uma frota gerenciada. EC2 A compilação será gravada em um diretório de saída, conforme indicado aqui: `out\linux\amd64\gamelift-servers-managed-ec2`.
2. Copie o executável do servidor do jogo e todos os arquivos associados necessários para execução na `gamelift-servers-managed-ec2` pasta. Você pode ter diretórios aninhados conforme necessário.

Um exemplo de estrutura de diretórios terá a seguinte aparência:

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

Etapa 3: configurar a embalagem para sua frota

Amazon GameLift Servers gerencia o ciclo de vida das instâncias computacionais de uma frota, criando novas instâncias com a criação do servidor instalada e reciclando instâncias conforme necessário. O serviço gerencia o ciclo de vida do processo do servidor de jogos que é executado em cada instância. Uma EC2 frota gerenciada pode ter instâncias em vários locais para oferecer suporte aos jogadores onde quer que estejam.

Edite o arquivo `config.yaml` para configurar o wrapper para registro, configuração de portas e inicialização do servidor.

1. Defina as configurações de registro. O invólucro do servidor de jogos gera mensagens de registro para cada processo do servidor de jogos. Por padrão, o nível de registro é definido para depurar para obter a máxima verbosidade. Isso é muito útil durante a configuração e a solução de problemas e determina o quão detalhadas são as mensagens de log - nesse caso, as mais detalhadas. As opções incluem depuração, informações, aviso e erro (menos detalhado).
2. Especifique o caminho para o diretório de log do servidor do jogo. O caminho padrão para os registros do servidor do jogo é `./game-server-logs`. Esse diretório contém todos os registros gerados pelo seu servidor de jogo e cada instância os tem. Os registros são enviados automaticamente para Amazon GameLift Servers onde são acessados na guia Eventos. Consulte a seção Solução de problemas para obter mais detalhes.
3. Defina a configuração da porta de rede. Defina a porta do jogo para o que você preferir. Para este tutorial, especifique apenas uma porta, pois você criará uma frota que executa somente um processo de servidor de jogo simultâneo por instância. Se você decidir executar vários processos ao mesmo tempo, precisará configurar portas suficientes para cada processo simultâneo. O valor padrão é 37016, conforme mostrado no arquivo de configuração, mas, em geral, para frotas que usam compilações Linux, use as portas 22 e 1026-60000. Para frotas que usam versões do Windows, use as portas 1026-60000.
4. Configure o caminho para o executável do servidor do jogo. Para `./MyGame/my-server-executable.exe` personalizar o caminho para o executável do servidor de jogos com o nome e a localização reais do mesmo. Este é o ponto de entrada para iniciar seu servidor de jogos.
5. Configure os argumentos do servidor do jogo. No mínimo, especifique um argumento `-port` e use o mesmo valor de porta do jogo que você definiu anteriormente. O valor "pos" 0 indica que esse é o primeiro argumento. Adicione outros argumentos conforme necessário. Esses argumentos são passados para o servidor do jogo quando ele é lançado, permitindo que você configure seu comportamento em tempo de execução.
 - a. Argumento: `--port`
 - b. Valor: `"{{.port number here}}"`
 - c. Posição: 0 (Primeiro argumento na lista)

Exemplo de configuração:

```
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
```

Etapa 4: faça o upload da versão do servidor do jogo

Agora você concluiu todos os elementos necessários da compilação do servidor de jogos (wrapper do servidor de jogos, config.yaml e arquivos do servidor de jogos) e está pronto para fazer o upload da compilação do jogo para hospedagem. Amazon GameLift Servers A maneira mais rápida de fazer o upload da versão do seu jogo é usando o AWS CLI comandoupload-build, conforme mostrado no exemplo a seguir.

Fazendo o upload de uma versão de jogo com o 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_2016 \
  --server-sdk-version 5.3.0 \
  --region us-west-2
```

Note

Para versões Mac e Linux, use --operating-system AMAZON_LINUX_2023

Ao criar a compilação, registre a ID da compilação da resposta da API para usá-la na criação da frota.

Etapa 5: criar a EC2 frota gerenciada

As etapas a seguir descrevem uma configuração mínima da frota, para que você possa começar a trabalhar o mais rápido possível.

Para criar sua frota:

1. Faça login no AWS Management Console e navegue até Amazon GameLift Servers.
2. Na barra de menu na parte superior da janela do console, verifique em qual região sua compilação está. Anote isso porque sua frota precisa estar na mesma região ou você não conseguirá encontrar ou escolher sua construção.
3. No painel de navegação na EC2 seção Gerenciado, escolha Construções.
4. Selecione a compilação que você enviou anteriormente para exibir a página de detalhes da compilação.
5. Na seção Frotas, escolha Criar frota, que exibe a página Definir detalhes da EC2 frota gerenciada, na qual você pode rastrear o status da sua frota, bem como visualizar os eventos de criação da frota na guia Eventos.
6. Preencha o nome e a descrição e escolha Avançar.
7. Na página Definir detalhes da instância, a região da compilação é mostrada por padrão. Escolha qualquer região adicional que você queira adicionar.
8. Para Tipo de frota, escolha On-Demand.
9. Em Tipos de instância, escolha c5.large e selecione Avançar.
10. Em Configuração de tempo de execução, como sua versão de jogo carregada usa o wrapper, você precisa apontar para o executável do wrapper. Para servidores de jogos Windows, isso é `C:\game\amazon-gamelift-servers-game-server-wrapper.exe`. Para servidores de jogos Linux, isso é `/local/game/amazon-gamelift-servers-game-server-wrapper`.

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

Além disso, configure os valores da porta do jogo com um intervalo que permita a porta que foi definida nos parâmetros de inicialização da configuração `config.yaml` e do tempo de execução. A porta de entrada `config.yaml` não precisa corresponder à especificada na configuração de tempo de execução da frota, mas em tempo de execução, se a configuração especificar uma porta diferente, esse valor substituirá o que está no `config.yaml`. Os parâmetros de inicialização inseridos na configuração do tempo de execução também substituem o que está no `config.yaml`.

11. Na página Revisar e criar, verifique novamente todas as configurações e escolha Enviar para criar sua frota. O status da frota mudará à medida que aumenta a capacidade de hospedar o servidor do jogo e, em breve, mostrará o status de Ativo. Quando a ativação estiver concluída e a frota

implantada, o serviço inicia o invólucro que está pronto para receber uma solicitação de sessão de jogo.

Etapa 6: criar e conectar-se a uma sessão de jogo

Quando o status da sua frota mostra Ativo, isso significa que os servidores do jogo estão prontos e aguardando para hospedar uma sessão de jogo. Para iniciar uma sessão de jogo, faça uma solicitação de sessão de jogo ao Amazon GameLift Servers serviço. Aqui, você usará o AWS CLI para fazer essa solicitação.

Note

Lembre-se de que criar uma sessão de jogo usando o AWS CLI é útil para testar e se familiarizar com o processo. Em algum momento, você adicionará chamadas programáticas de AWS SDK ao serviço de back-end do jogo como parte do sistema de matchmaking ou posicionamento de sessões de jogo do seu jogo.

Use o seguinte para criar uma sessão de jogo:

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

Você também pode passar propriedades personalizadas do jogo para o executável do servidor. Consulte os argumentos do servidor de jogo no README para obter detalhes. Ao receber a create-game-session chamada, eles Amazon GameLift Servers informam o wrapper para iniciar o executável do servidor do jogo e iniciar uma sessão de jogo. O que está nele config.yaml afeta a configuração do servidor do jogo e os parâmetros de inicialização definidos no console determinam a configuração da sessão de jogo em si.

Exemplo de formato para adicionar propriedades do jogo:

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

Etapa 7: Gerencie e monitore sua frota

Agora que sua frota de servidores de jogos está configurada e uma sessão de jogo foi iniciada, você pode gerenciá-la e monitorá-la no Amazon GameLift Servers console. A melhor maneira de fazer isso é na página de detalhes da frota, onde você pode editar os detalhes da frota ou alterar o dimensionamento e a capacidade da frota na guia Escalonamento. Consulte a seção a seguir sobre como escalar seus servidores de jogos.

Escolha a guia Métricas para visualizar gráficos que descrevem o monitoramento das métricas de atividade e hardware. Para obter detalhes sobre os gráficos de métricas, escolha o link Informações ao lado do ID da sua frota. Além disso, embora você possa monitorar de perto seus servidores de jogos na guia Métricas, você também pode adicionar alarmes a essas métricas no CloudWatch painel.

Para navegar até o CloudWatch painel a partir do console:

1. Digite "CloudWatch" na barra de pesquisa e escolha-a na lista de resultados da pesquisa para exibir a CloudWatch Visão geral.
2. Role para baixo e escolha Exibir GameLift painel para ver gráficos com base nas principais métricas de suas frotas e sessões de jogo.

Etapa 8: escale seus servidores de jogos

Para a próxima etapa, você configura o escalonamento automático. Com o escalonamento automático, você escala dinamicamente a capacidade da sua frota em resposta à atividade do servidor do jogo. Conforme os jogadores chegam e iniciam as sessões de jogo, o escalonamento automático adiciona mais instâncias e, à medida que a demanda dos jogadores diminui, o

escalonamento automático libera instâncias não utilizadas. Essa é uma forma eficaz de minimizar seus recursos e custos de hospedagem e, ao mesmo tempo, proporcionar uma experiência de jogador tranquila e rápida.

Ao se preparar para o lançamento do jogo, você deve configurar o escalonamento automático para suas frotas. O escalonamento automático é recomendado como uma forma eficaz de minimizar seus recursos e custos de hospedagem e, ao mesmo tempo, proporcionar uma experiência de jogador tranquila e rápida.

Para configurar manualmente a capacidade da frota

Para configurar manualmente a capacidade da frota

1. Acesse a guia Escalonamento na página de detalhes da sua frota.
2. Selecione um local e escolha Editar.
3. Altere o valor das instâncias desejadas e altere as configurações de tamanho mínimo e máximo para escalar além dos valores atuais.

Note

Use a configuração máxima de instâncias como uma brecha para evitar custos e escalabilidade excessivos.

4. Escolha Confirmar.

Para usar o escalonamento automático baseado em metas

Para usar o escalonamento automático baseado em metas

O escalonamento automático baseado em metas (rastreamento de metas) vincula o escalonamento da frota à porcentagem de sessões de jogo disponíveis. À medida que os jogadores começam a jogar seu jogo e as sessões de jogo disponíveis diminuem, ele responde adicionando automaticamente mais instâncias à frota.

1. Em Política de escalonamento automático baseada em metas, escolha Adicionar política e defina a capacidade da frota de mudar automaticamente quando atingir o limite da porcentagem de sessões de jogo disponíveis que você definiu. Um buffer maior pode lidar melhor com surtos,

fazendo com que novos jogadores entrem nos jogos rapidamente, mas também pode aumentar os custos de hospedagem.

2. Escolha Confirmar para aceitar as alterações.

O escalonamento automático baseado em regras oferece um controle mais granular, como a capacidade de vincular o escalonamento a outras métricas da frota e definir limites personalizados e respostas de escalabilidade. Ele oferece opções poderosas, mas também requer o uso da CLI e testes significativos para entender como as regras personalizadas se comportam em ação. Este tutorial se concentra em primeiro configurar a abordagem baseada em metas.

Solução de problemas comuns

A seguir estão os problemas comuns que você pode encontrar ao executar seus servidores e sessões de jogo. Quando o servidor ou a sessão do jogo não estão funcionando corretamente, a primeira etapa é verificar os registros, o que pode revelar um dos problemas descritos abaixo para novas implantações ou jogos em produção.

O seguinte geralmente é revelado nos registros:

- O processo do servidor do jogo não pode ser iniciado. Isso pode ser um erro na configuração do wrapper. Verifique se o arquivo tem o caminho de inicialização correto e os parâmetros e argumentos de inicialização corretos.
- A compilação do servidor de jogos não pode ser executada. Provavelmente é um erro no código do jogo.
- Os jogadores não conseguem se conectar às sessões de jogo. Provavelmente é um erro de configuração da porta.
- Conexões atrasadas ou lentas. Analise as políticas e os limites de escalabilidade.
- Sem conexão. Verifique as regras e a configuração portuária da sua frota.

Para ver os registros de eventos da sua Amazon GameLift Servers frota

Para ver os registros de eventos da sua Amazon GameLift Servers frota

1. Abra o console do Amazon GameLift Servers.
2. Na página Detalhes da frota, escolha a guia Eventos e baixe o registro. Você também pode monitorar métricas de atividade e hardware para a integridade do servidor de jogos e ativações de sessões de jogos na guia Métricas.

Para ver os registros da sessão do jogo

Para ver os registros da sessão do jogo

1. No console, abra sua frota e abra a guia Sessões do jogo.
2. Escolha um ID de sessão de jogo na lista para exibir a página de visão geral.
3. Escolha Baixar registros para baixar um arquivo de log localmente.

Para ver os registros da sessão do jogo com a CLI, use a `GetGameSessionLogURL` API. Amazon GameLift Servers armazena automaticamente os registros por 14 dias.

Você também pode configurar CloudWatch os registros da Amazon para sua frota. Isso fornece recursos adicionais de registro e integração com outros serviços AWS de monitoramento.

Para acesso aos registros em tempo real ou períodos de retenção estendidos por meio de CloudWatch:

1. Na parte superior do painel do Amazon GameLift Servers console, digite "CloudWatch" na barra de pesquisa e escolha-a no menu suspenso de resultados.
2. Acesse os grupos de CloudWatch registros e pesquise sessões específicas. O método mais fácil é clicar em Pesquisar tudo e filtrar usando `gameSessionId` nosso `ClientID`.

Próximas etapas

- [Crie uma frota com vários locais para adicionar hospedagem em outros locais](#)
- [Adicione uma fila de sessões de jogo para fazer os melhores posicionamentos possíveis para sessões de jogo em vários locais](#)
- [Experimente o FlexMatch matchmaking criando um matchmaker e um conjunto de regras para o seu jogo](#)
- [Comece a trabalhar na funcionalidade do cliente do jogo e dos componentes do serviço de back-end, para que os jogadores possam fazer solicitações de participação e se conectar diretamente às sessões do jogo](#)
- [Quando estiver pronto, mude para uma solução totalmente integrada](#)

Explore com o Amazon GameLift Servers plug-in

A ferramenta Amazon GameLift Servers O plugin é um complemento completo para o seu motor de jogo Unreal ou Unity. Ele orienta você nas etapas básicas para implantar seu jogo para hospedagem com Amazon GameLift Servers. Com o conjunto de ferramentas e os fluxos de trabalho do plug-in, você pode trabalhar em seu ambiente de desenvolvimento de mecanismos de jogos para preparar seu servidor de jogos para hospedagem, configurar a hospedagem em uma máquina local para testes, criar um serviço de back-end simples e implantar seu servidor de jogos em hospedagem gerenciada baseada em nuvem.

Use o plug-in para experimentar o trabalho com Amazon GameLift Servers e obtenha uma solução de hospedagem de jogos em funcionamento rapidamente. Você pode trabalhar com exemplos de ativos do jogo ou com seu próprio projeto de jogo. O plug-in automatiza várias etapas para que você possa criar rapidamente uma solução de trabalho simples. Ao concluir os fluxos de trabalho guiados do plug-in, você poderá conectar um cliente de jogo a sessões de jogo hospedadas ao vivo por meio de Amazon GameLift Servers. Depois de usar o plug-in para criar uma solução de hospedagem simples, você pode personalizar sua solução para atender às necessidades do seu jogo.

O plug-in está disponível para os seguintes mecanismos de jogos:

- Unreal Engine
- Unity

O plug-in inclui os seguintes componentes para cada mecanismo de jogos:

- Módulos de plug-in para o editor do mecanismo de jogos. Quando o plug-in é instalado, um novo botão do menu principal dá acesso a Amazon GameLift Servers funcionalidade.
- Bibliotecas para o Amazon GameLift Servers API de serviço com funcionalidade do lado do cliente.
- Bibliotecas para o Amazon GameLift Servers SDK do servidor (versão 5).
- Exemplos de ativos para uso no teste de uma integração de servidor.
- Configurações editáveis, na forma de AWS CloudFormation modelos, que definem sua solução de servidor de jogos.

Tópicos

- [Fluxo do plug-in](#)
- [Amazon GameLift Serversplugin para Unreal Engine](#)

- [Amazon GameLift Serversplugin para Unity \(servidor SDK 5.x\)](#)
- [Amazon GameLift Serversplugin para Unity \(servidor SDK 4.x\)](#)

Fluxo do plug-in

As etapas a seguir descrevem um caminho típico para preparar e implantar seu projeto de jogo no Amazon GameLift Servers. Você conclui essas etapas trabalhando no editor do mecanismo de jogo e no código do jogo.

1. Crie um perfil de usuário vinculado ao usuário AWS da sua conta e forneça credenciais de acesso com permissões de uso Amazon GameLift Servers.
2. Configure AWS os recursos relacionados que o plug-in usa na solução de hospedagem (chamados de “bootstrapping”).
3. Adicione o código do servidor ao seu projeto para estabelecer comunicação entre um servidor de jogo em execução e o Amazon GameLift Servers serviço.
4. Adicione o código do cliente ao seu projeto para permitir que os clientes do jogo enviem solicitações para Amazon GameLift Servers para iniciar novas sessões de jogo e depois se conectar a elas.
5. Use o fluxo de trabalho Anywhere para configurar a estação de trabalho local como um recurso de computação Anywhere e hospedar o servidor de jogos. Inicie o servidor e o cliente de jogos localmente por meio do plug-in, conecte-se a uma sessão de jogo e teste a integração.
6. Use o EC2 fluxo de trabalho gerenciado para carregar seu servidor de jogos para Amazon GameLift Servers e implante uma solução de hospedagem em nuvem simples, mas completa. Inicie o cliente de jogo localmente por meio do plug-in, solicite uma sessão de jogo, conecte-se a ela e jogue.

Ao trabalhar no plug-in, você criará e usará AWS recursos. Essas ações podem incorrer em cobranças na AWS conta em uso. Se você é novato AWS, essas ações podem estar cobertas pelo [nível AWS gratuito](#).

Amazon GameLift Serversplugin para Unreal Engine

Esse plug-in adiciona o SDK e as ferramentas do servidor Amazon GameLift Servers C++ ao editor UE. Use os fluxos de trabalho guiados da interface do usuário para integrar a funcionalidade do SDK do servidor em seu projeto de jogo e implantar uma solução de Amazon GameLift Servers hospedagem para seu servidor de jogos.

Com o plug-in, você pode criar uma solução básica de hospedagem funcional e, em seguida, otimizar e personalizar conforme necessário. Configure uma frota Amazon GameLift Servers Anywhere com sua estação de trabalho local como anfitriã. Para hospedagem na nuvem com frotas de contêineres gerenciadas EC2 ou gerenciadas, implante seu servidor de jogos com uma solução completa para gerenciar solicitações de sessões de jogos e conexões de clientes.

Tópicos

- [Instale o plug-in para seu projeto de jogo Unreal](#)
- [Próximas etapas: personalize sua solução de hospedagem de jogos](#)
- [Plugin para Unreal: configure um perfil de AWS usuário](#)
- [Plugin para Unreal: integre seu código de jogo](#)
- [Plugin para Unreal: hospede seu jogo localmente com Anywhere Amazon GameLift Servers](#)
- [Plugin para Unreal: implante seu jogo em uma frota gerenciada EC2](#)
- [Plugin para Unreal: implante seu jogo em uma frota de contêineres gerenciada](#)

Instale o plug-in para seu projeto de jogo Unreal

[Obtenha o Amazon GameLift Servers plug-in para o Unreal Engine em GitHub](#)

Consulte o readme do GitHub repositório para obter informações sobre como instalar o plug-in no seu Unreal Editor para um projeto de jogo.

O plug-in inclui os seguintes componentes:

- Módulos de plug-in para o editor do UE. Quando o plug-in é instalado, um novo botão do menu principal dá acesso à Amazon GameLift Servers funcionalidade.
- Bibliotecas C++ para a API Amazon GameLift Servers de serviço. Use a funcionalidade de API em um serviço de back-end do lado do cliente para ajudar clientes de jogos a solicitar sessões e informações sobre sessões de send/retrieve jogos.
- Bibliotecas irreais para o SDK Amazon GameLift Servers do servidor (versão 5). Use o SDK do servidor no código do servidor do jogo para gerenciar a comunicação entre os processos do servidor de jogos hospedado e o Amazon GameLift Servers serviço.
- Conteúdo para teste, incluindo um mapa do jogo de startup e dois mapas de teste com esquemas básicos e elementos de interface do usuário para uso no teste de uma integração de servidor.
- Configurações editáveis, na forma de AWS CloudFormation modelos, que o plug-in usa ao implantar seu servidor de jogos para hospedagem.

Esse plug-in usa modelos AWS CloudFormation para implantar soluções de hospedagem para cenários comuns de jogos. Use essas soluções conforme fornecidas ou personalize-as conforme necessário para seus jogos.

Próximas etapas: personalize sua solução de hospedagem de jogos

Usar os fluxos de trabalho guiados do plug-in é uma boa maneira de começar a usar rapidamente uma solução de Amazon GameLift Servers hospedagem. Com o plug-in, você pode configurar versões básicas de cada um dos componentes da sua solução.

Quando estiver pronto, você poderá desenvolver essa solução básica personalizando cada componente e ajustando sua solução enquanto se prepara para o lançamento do jogo. Considere estas opções:

- Modifique suas frotas e configurações de frota. Consulte [Personalize seu Amazon GameLift Servers EC2 frotas gerenciadas](#).
- Personalize a configuração da fila da sessão de jogo. Veja [Personalize uma fila de sessões de jogo](#):
- Adicione funcionalidade ao seu servidor de jogos e cliente de jogos. Consulte [Integre seu servidor de jogos com Amazon GameLift Servers](#) e [Integre seu cliente de jogo com Amazon GameLift Servers](#).
- Personalize seu serviço de back-end. Consulte [Projete seu serviço de cliente de jogos](#).
- Configure o ajuste de escala automático da capacidade para atender à demanda esperada dos jogadores. Consulte [Escalando a capacidade de hospedagem de jogos com Amazon GameLift Servers](#).
- Configure ferramentas de observabilidade de hospedagem, incluindo análises e registros em log. Consulte [Monitoramento Amazon GameLift Servers](#).
- Automatize a implantação usando [infraestrutura como código \(IaC\)](#). Os fluxos de trabalho guiados do plug-in para soluções gerenciadas usam AWS CloudFormation modelos. Você pode personalizá-los conforme necessário. Consulte [Gerenciamento Amazon GameLift Servers recursos de hospedagem usando AWS CloudFormation](#).

Tópicos

- [Plugin para Unreal: configure um perfil de AWS usuário](#)
- [Plugin para Unreal: integre seu código de jogo](#)
- [Plugin para Unreal: hospede seu jogo localmente com Anywhere Amazon GameLift Servers](#)

- [Plugin para Unreal: implante seu jogo em uma frota gerenciada EC2](#)
- [Plugin para Unreal: implante seu jogo em uma frota de contêineres gerenciada](#)

Plugin para Unreal: configure um perfil de AWS usuário

Depois de instalar o plug-in, configure um perfil de usuário com uma AWS conta válida. Você pode manter vários perfis no plug-in, mas só pode selecionar um perfil por vez. Sempre que você trabalha no plug-in, selecione um perfil para usar. Cada página do fluxo de trabalho exibe o perfil atualmente selecionado.

A manutenção de vários perfis permite alternar entre diferentes implantações de hospedagem. Por exemplo, você pode configurar perfis que usam a mesma AWS conta, mas implantados em AWS regiões diferentes. Ou você pode configurar perfis com AWS contas ou usuários e conjuntos de permissões diferentes.

Note

Se você instalou a AWS CLI em sua estação de trabalho e já tem um perfil configurado, o Amazon GameLift Servers plug-in o detectará e o listará como um perfil existente. O plug-in seleciona automaticamente qualquer perfil nomeado [default]. Você poderá usar um perfil existente ou criar outro.

Todos os perfis devem ser inicializados para configurar alguns AWS recursos necessários em sua conta de usuário.

Para gerenciar seus AWS perfis

1. Na barra de ferramentas principal do editor Unreal, escolha o Amazon GameLift Servers menu e selecione Credenciais de AWS acesso. Essa ação abre o Amazon GameLift Servers plug-in na página Configurar seu perfil de usuário.
2. Use os botões para criar uma nova AWS conta ou configurar um perfil de usuário para uma AWS conta que você já tem.
3. Se você ainda não tiver um perfil de usuário, será solicitado que você insira os detalhes do perfil e crie um novo perfil. Forneça as informações a seguir:
 - Uma AWS conta. Se você criar uma nova AWS conta, use o link para o AWS Management Console e siga as instruções. Consulte [Criar uma conta da AWS](#) para obter mais detalhes.

- Um AWS usuário com permissões de uso Amazon GameLift Servers e outros AWS serviços necessários. Consulte [Configurar um Conta da AWS](#) para obter instruções sobre como configurar um usuário AWS Identity and Access Management (IAM) com Amazon GameLift Servers permissões e acesso programático com credenciais de longo prazo.
 - Credenciais para seu AWS usuário. Essas credenciais consistem em um ID de chave de AWS acesso e uma chave AWS secreta. Consulte [Obter suas chaves de acesso](#) para obter mais detalhes.
 - AWS região. Essa é uma localização geográfica em que você deseja criar seus recursos de hospedagem da AWS . Durante o desenvolvimento, recomendamos usar uma região próxima à sua localização física. Escolha uma região na lista de [AWS regiões suportadas](#).
4. Se o plug-in detectar perfis existentes, ele exibirá uma lista dos perfis disponíveis. Selecione um perfil existente na lista ou escolha Adicionar outro perfil para criar um novo.

Inicialize um perfil de usuário

Todos os perfis devem ser inicializados para serem usados com o plug-in. Amazon GameLift Servers O Bootstrapping cria um bucket do Amazon S3 específico para o perfil. Ele é usado para armazenar configurações de projetos, compilar artefatos e outras dependências. Os buckets não são compartilhados entre outros perfis.

O bootstrapping envolve a criação de novos AWS recursos e pode incorrer em custos.

Para inicializar seu perfil:

1. Na página Credenciais de AWS acesso, verifique o status de bootstrap do perfil de usuário que você deseja usar. Se o status de bootstrap do perfil for "Inativo" e não houver nenhum bucket do S3 listado, você precisará inicializar o perfil.
2. Selecione o perfil que você deseja usar e escolha o perfil Bootstrap.
3. Aguarde até que o status do bootstrap mude para "Active". Isso pode levar alguns minutos.

Plugin para Unreal: integre seu código de jogo

Antes de implantar seu servidor de jogos em uma frota, você precisa fazer uma série de atualizações no código do jogo e empacotar os componentes do jogo para uso com o Amazon GameLift Servers serviço.

Este tópico explica as etapas para fazer uma integração mínima. Para integração com o servidor, use a amostra de código fornecida para atualizar o modo de jogo do seu projeto.

- [Configure metas de construção e regras de módulo](#)
- [Atualize o código do seu servidor de jogos](#)
- [Integre o mapa de jogo do seu cliente](#)
- [Empacotar os componentes de jogo](#)

Configure metas de construção e regras de módulo

Modifique seus arquivos de projeto de jogo para gerar corretamente os componentes de compilação para uso com Amazon GameLift Servers.

Para adicionar destinos de compilação de cliente e servidor:

1. Abra os arquivos de código do seu projeto de jogo e localize o `.../Games/[your application name]Source/[your application name]Target.cs` arquivo. Exemplo: `.../Source/GameLiftUnrealAppTarget.cs`. (Se você usa o Visual Studio, abra o `.sln` arquivo do projeto.)
2. Copie esse arquivo para criar dois novos arquivos de destino no `Source/` diretório.
 - Destino do cliente — Renomeie o novo arquivo para `[your application name]Client.Target.cs`. Edite o conteúdo para atualizar os valores do nome da classe e do tipo de destino, conforme ilustrado no código de exemplo a seguir:

```
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" );
    }
}
```

- Destino do servidor — Renomeie o novo arquivo para *[your application name]*Server.Target.cs. Edite o conteúdo para atualizar os valores do nome da classe e do tipo de destino, conforme ilustrado no código de exemplo a seguir:

```
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. Regenere seus arquivos de projeto. Se você estiver usando o Visual Studio, você pode clicar com o botão direito do mouse no .uproject arquivo do seu projeto de jogo e selecionar Gerar arquivos de projeto do Visual Studio.

Para atualizar as regras do módulo do projeto do jogo:

Atualize as regras do módulo do projeto do jogo para depender do plug-in.

1. Abra os arquivos de código do seu projeto de jogo e localize o .../Games/*[your application name]*Source/*[your application name]*.Build.cs arquivo. Exemplo: .../Source/GameLiftUnrealApp.Build.cs. (Se você usa o Visual Studio, abra o .sln arquivo do projeto.)
2. Localize a ModuleRules classe e atualize conforme ilustrado no código de exemplo a seguir:

```
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",
"GameLiftServerSDK" });

        bEnableExceptions = true;
    }
}

```

3. Depois de criar os novos arquivos de destino e modificar as regras do módulo, reconstrua seu projeto de jogo.

Atualize o código do seu servidor de jogos

Atualize o código do servidor de jogos para permitir a comunicação entre um processo do servidor de jogos e o Amazon GameLift Servers serviço. Seu servidor de jogo deve ser capaz de responder às solicitações de Amazon GameLift Servers, como iniciar e interromper novas sessões de jogo.

Para adicionar o código do servidor para Amazon GameLift Servers

1. No editor de código, abra o arquivo de solução (.sln) do seu projeto de jogo, normalmente encontrado na pasta raiz do projeto. Por exemplo: GameLiftUnrealApp.sln.
2. Com a solução aberta, localize o arquivo de cabeçalho do modo de jogo do projeto: arquivo [project-name]GameMode.h. Por exemplo: GameLiftUnrealAppGameMode.h.
3. Altere o arquivo de cabeçalho para alinhá-lo com o código a seguir. Certifique-se de substituir "GameLiftServer" pelo nome do seu próprio projeto. Essas atualizações são específicas para o servidor do jogo, recomendamos que você faça uma cópia de backup dos arquivos originais do modo de jogo para uso com seu cliente.

Exemplo de código 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;
};

```

- Abra o arquivo [project-name]GameMode.cpp de origem relacionado (por exemplo GameLiftUnrealAppGameMode.cpp). Altere o código para alinhá-lo com o código de exemplo a seguir. Certifique-se de substituir "GameLiftUnrealApp" pelo nome do seu próprio projeto. Essas atualizações são específicas para o servidor do jogo, recomendamos que você faça uma cópia de backup do arquivo original para uso com seu cliente.

O código de exemplo a seguir mostra como adicionar os elementos mínimos necessários para a integração do servidor com Amazon GameLift Servers:

- Inicialize um cliente de Amazon GameLift Servers API. A `InitSDK()` chamada com os parâmetros do servidor é necessária para uma frota Amazon GameLift Servers Anywhere. Quando você se conecta a uma frota Anywhere, o plug-in armazena os parâmetros do servidor como argumentos do console. O código de amostra pode acessar os valores em runtime.
- Implemente as funções de retorno de chamada necessárias para responder às solicitações do Amazon GameLift Servers serviço `OnStartGameSession`, incluindo `OnProcessTerminate`, e. `onHealthCheck`

- Ligue `ProcessReady()` para uma porta designada para notificar o Amazon GameLift Servers serviço quando estiver pronto para hospedar sessões de jogo.

Exemplo de código de servidor de jogo

```
// 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);
        }
    }
#endif
}

```

```

        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"));
        GameLiftSdkModule->ProcessEnding();
    });

//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("GameServerLog/Saved/Logs/GameServerLog.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
}

```

Integre o mapa de jogo do seu cliente

O mapa do jogo de startup contém elementos lógicos e de interface do usuário do Blueprint que já incluem código básico para solicitar sessões de jogo e usar informações de conexão para se conectar a uma sessão de jogo. Você poderá usar o mapa como está ou modificá-lo conforme necessário. Use o mapa do jogo de startup com outros ativos do jogo, como o projeto modelo de terceira pessoa fornecido pelo Unreal Engine. Esses ativos estão disponíveis no Content Browser. Você poderá usá-los para testar os fluxos de trabalho de implantação do plug-in ou como um guia para criar um serviço de backend personalizado para o jogo.

O mapa de startup de arquivo tem as seguintes características:

- Inclui lógica tanto para uma frota Anywhere quanto para uma EC2 frota gerenciada. Ao administrar seu cliente, você pode optar por se conectar a qualquer frota.
- A funcionalidade do cliente inclui encontrar uma sessão de jogo (`SearchGameSessions()`), criar uma sessão de jogo (`CreateGameSession()`) e participar diretamente de uma sessão de jogo.
- Ele obtém um ID de jogador exclusivo do grupo de usuários do Amazon Cognito do seu projeto (isso faz parte de uma solução implantada em qualquer lugar).

Para usar o mapa do jogo de startup

1. No editor do UE, abra a página Configurações do projeto, mapas e modos e expanda a seção Mapas padrão.
2. Para Editor Startup Map, selecione `StartupMap ""` na lista suspensa. Talvez seja necessário pesquisar o arquivo, que está localizado em `... > Unreal Projects/[project-name]/Plugins/Amazon GameLift Servers Plugin Content/Maps`.
3. Para Mapa padrão do jogo, selecione o mesmo "StartupMap" na lista suspensa.
4. Para Mapa padrão do servidor, selecione "ThirdPersonMap". Esse é um mapa padrão incluído no seu projeto de jogo. Este mapa foi projetado para dois jogadores no jogo.
5. Abra o painel de detalhes do mapa padrão do servidor. Defina GameMode Override como "Nenhum".
6. Expanda a seção Modos padrão e defina o Modo de jogo padrão global do servidor para o modo de jogo que você atualizou para a integração do servidor.

Depois de fazer essas alterações no projeto, você estará pronto para criar os componentes do jogo.

Empacotar os componentes de jogo

Como empacotar as compilações de servidor e cliente de jogo

1. Abra seu projeto de jogo em uma versão original do editor do Unreal Engine.
2. Use o editor para empacotar suas versões de cliente e servidor do jogo.
 - a. Escolha um destino. Vá para Plataformas, Windows e selecione uma das seguintes opções:
 - Servidor: `[your-application-name]Server`
 - Cliente: `[your-application-name]Client`
 - b. Inicie a compilação. Vá para Plataforma, Windows, Projeto de Pacote.

Cada processo de empacotamento gera um executável: `[your-application-name]Client.exe` ou `[your-application-name]Server.exe`.

No plug-in, defina os caminhos para os executáveis de criação do cliente e do servidor em sua estação de trabalho local.

Plugin para Unreal: hospede seu jogo localmente com Anywhere Amazon GameLift Servers

Use esse fluxo de trabalho para configurar sua estação de trabalho local como host de servidor de jogos usando uma frota Anywhere. Você pode usá-lo para testar a integração do servidor de jogos antes de implantar em uma frota gerenciada baseada em nuvem. Também pode ser útil para testes locais durante o desenvolvimento iterativo de jogos.

Para iniciar o fluxo de trabalho Amazon GameLift Servers Anywhere:

- Na barra de ferramentas principal do editor Unreal, escolha o Amazon GameLift Servers menu e selecione Hospedar com qualquer lugar. Essa ação abre a página do plug-in Implantar o Anywhere, que apresenta um processo de seis etapas para integrar, criar e iniciar os componentes do jogo.

Etapa 1: configurar seu perfil

Escolha o perfil que você deseja usar ao seguir esse fluxo de trabalho. O perfil selecionado afeta todas as etapas do fluxo de trabalho. Todos os recursos que você cria estão associados à AWS conta do perfil e são colocados na AWS região padrão do perfil. As permissões do usuário do perfil determinam seu acesso aos AWS recursos e ações.

Como definir um perfil do usuário

1. Selecione um perfil na lista suspensa de perfis disponíveis. Se você ainda não tem um perfil ou deseja criar um novo, acesse o GameLift menu Amazon e escolha Definir perfis de AWS usuário.
2. Se o status do bootstrap não for “Ativo”, escolha o perfil do Bootstrap e espere que o status mude para “Ativo”.

Etapa 2: configurar o código do jogo

Nesta etapa, prepare as versões do servidor e do cliente do jogo para trabalhar com Amazon GameLift Servers elas. Se você ainda não integrou o código do jogo, consulte [Plugin para Unreal: integre seu código de jogo](#). Insira os caminhos para os executáveis do jogo em sua estação de trabalho local.

- Servidor de jogos: integre seu servidor de jogos com o SDK do servidor Amazon GameLift Servers e empacote sua compilação de servidor de jogos. Para instruções, consulte [Plugin para Unreal: integre seu código de jogo](#). O servidor do jogo deve estar integrado ao SDK do servidor para estabelecer comunicação com o Amazon GameLift Servers serviço e responder às solicitações para iniciar novas sessões de jogo e aceitar conexões do cliente do jogo.
- Cliente de jogo: no mínimo, você precisa de um cliente de jogo que possa se conectar ao seu servidor de jogo usando endereço IP e informações de porta. Se você ainda não tem os componentes do seu cliente de jogo configurados Amazon GameLift Servers, você pode usar a AWS CLI ferramenta para solicitar manualmente novas sessões de jogo, obter informações de conexão e usar essas informações para conectar o cliente do jogo.

Em algum momento, você precisará ter um serviço de back-end para enviar novas solicitações de sessões de jogo ao Amazon GameLift Servers serviço e retransmitir as informações de conexão para um cliente do jogo. Você pode usar os mapas de teste incluídos no plug-in para adicionar a Amazon GameLift Servers funcionalidade do cliente ao seu projeto de jogo. Para obter ajuda na criação de uma solução personalizada, consulte [Adicione Amazon GameLift Servers ao seu cliente de jogo](#).

Etapa 3: conectar-se a uma frota do Anywhere

Nesta etapa, você designa uma frota do Anywhere para usar. Uma frota do Anywhere define uma coleção de recursos computacionais, que podem estar localizados em qualquer lugar, para hospedagem de servidores de jogos.

- Se a AWS conta que você está usando atualmente tiver frotas Anywhere existentes, abra o campo suspenso Nome da frota e escolha uma frota. Essa lista suspensa mostra apenas as frotas de Anywhere na AWS região para o perfil de usuário atualmente ativo.
- Se não houver frotas existentes, ou se você quiser criar uma, escolha Criar frota Anywhere e forneça um nome para a frota.

Depois de escolher uma frota Anywhere para seu projeto, Amazon GameLift Servers verifique se o status da frota está ativo e exibe o ID da frota. Você poderá acompanhar o progresso dessa solicitação no log de saída do editor do Unreal.

Etapa 4: registrar sua estação de trabalho

Nesta etapa, você registra sua estação de trabalho local como um recurso computacional na nova frota do Anywhere.

Como registrar a estação de trabalho como uma computação do Anywhere

1. Insira um nome de computação para a sua máquina local. Se você adicionar mais de uma computação na frota, os nomes deverão ser exclusivos.
2. Forneça um endereço IP para sua máquina local. Esse campo é padronizado para o endereço IP público da máquina. Você também pode usar o localhost (127.0.0.1), desde que esteja executando o cliente e o servidor do jogo na mesma máquina.
3. Escolha Registrar computação. Você poderá acompanhar o progresso dessa solicitação no log de saída do editor do Unreal.

Em resposta a essa ação, Amazon GameLift Servers verifica se ela pode se conectar à computação e retorna informações sobre a computação recém-registrada. Ele também cria os argumentos do console que os executáveis do seu jogo precisam ao inicializar a comunicação com o Amazon GameLift Servers serviço.

Etapa 5: gerar token de autenticação

Os processos do servidor de jogos que estão sendo executados em sua computação Anywhere precisam de um token de autenticação para fazer chamadas para o Amazon GameLift Servers serviço. O plug-in gera e armazena automaticamente um token de autenticação para a frota de Anywhere sempre que você inicia o servidor do jogo a partir do plug-in. O valor do token de autenticação é armazenado como um argumento de linha de comando, que o código do servidor pode recuperar em runtime.

Os exemplos de código fornecidos acima também permitem que você use o [AWS Signature Version 4 \(SigV4\) para solicitações de API](#). O SigV4 é o protocolo de AWS assinatura para adicionar informações de autenticação às solicitações da API.

Você não precisa realizar nenhuma ação nesta etapa.

Etapa 6: iniciar o jogo

Neste ponto, você concluiu todas as tarefas necessárias para iniciar e jogar seu jogo multijogador em uma estação de trabalho local usando Amazon GameLift Servers.

Como jogar o jogo hospedado

1. Inicie seu servidor de jogos. O servidor do jogo notificará Amazon GameLift Servers quando estiver pronto para hospedar as sessões do jogo.
2. Inicie seu cliente de jogo e use a nova funcionalidade para iniciar uma nova sessão de jogo. Essa solicitação é enviada Amazon GameLift Servers por meio do novo serviço de back-end. Em resposta, Amazon GameLift Servers, chama o servidor do jogo, em execução na sua máquina local, para iniciar uma nova sessão de jogo. Quando a sessão de jogo estiver pronta para aceitar jogadores, Amazon GameLift Servers fornece informações de conexão para que o cliente do jogo participe da sessão de jogo.

Plugin para Unreal: implante seu jogo em uma frota gerenciada EC2

Nesse fluxo de trabalho, implante seu jogo para hospedagem em recursos computacionais baseados em nuvem gerenciados pelo Amazon GameLift Servers. Faça o upload da versão integrada do servidor de jogos para o Amazon GameLift Servers serviço para implantação. Se você ainda não integrou o código do jogo, consulte [Plugin para Unreal: integre seu código de jogo](#). Quando esse fluxo de trabalho estiver concluído, você terá um cliente de jogo funcional que poderá se conectar aos servidores de jogos na nuvem.

Para iniciar o EC2 fluxo de trabalho Amazon GameLift Servers gerenciado da Amazon:

- Na barra de ferramentas principal do editor Unreal, escolha o Amazon GameLift Servers menu e selecione Host with Managed. EC2 Essa ação abre a página do plug-in Deploy Amazon EC2 Fleet, que apresenta um processo de seis etapas para integrar, criar, implantar e lançar os componentes do seu jogo.

Etapa 1: configure o perfil

Escolha o perfil que você deseja usar ao seguir esse fluxo de trabalho. O perfil selecionado afeta todas as etapas do fluxo de trabalho. Todos os recursos que você cria estão associados à AWS conta do perfil e são colocados na AWS região padrão do perfil. As permissões do usuário do perfil determinam seu acesso aos AWS recursos e ações.

Como definir um perfil do usuário

1. Selecione um perfil na lista suspensa de perfis disponíveis. Se você ainda não tem um perfil ou deseja criar um novo, acesse o GameLift menu Amazon e escolha Definir perfis de AWS usuário.
2. Se o status do bootstrap não for “Ativo”, escolha o perfil do Bootstrap e espere que o status mude para “Ativo”.

Etapa 2: configurar o código do jogo

Nesta etapa, prepare as compilações do servidor e do cliente do jogo para funcionar com o SDK do servidor Amazon GameLift Servers C++ para Unreal. Se você ainda não integrou o código do jogo e criou executáveis de cliente e servidor de jogos, consulte [Plugin para Unreal: integre seu código de jogo](#). Insira os caminhos para os executáveis do jogo em sua estação de trabalho local.

Nesta etapa do fluxo de trabalho, o plug-in fornece links para instruções e código-fonte para configurar uma versão original do Unreal Editor. Você precisa usar a versão de origem ao criar seus componentes de cliente e servidor.

Depois de criar um servidor de jogos integrado ao SDK do servidor, conclua as tarefas a seguir para prepará-lo Amazon GameLift Servers para envio e hospedagem.

Para preparar sua compilação de servidor para implantação na nuvem (Windows)

Na **WindowsServer** pasta em que o editor Unreal armazena os arquivos de compilação do servidor por padrão, faça as seguintes adições:

1. Copie o script de instalação da compilação do servidor na raiz da **WindowsServer** pasta. O script de instalação está incluído no download do plug-in. Procure o arquivo[project-name]/Plugins/Resources/CloudFormation/extra_server_resources/install.bat. Amazon GameLift Servers usa esse arquivo para instalar o servidor criado em seus computadores de hospedagem.
2. Copie o **VC_redist.x64.exe** arquivo na raiz da **WindowsServer** pasta. Esse arquivo está incluído na instalação do Visual Studio. Geralmente está localizado em C:/Program Files (x86)/Microsoft Visual Studio/2019/Professional/VC/Redist/MSVC/v142.
3. Adicione os arquivos da biblioteca OpenSSL à compilação do seu servidor de jogos.

 Note

Se o seu servidor de jogos estiver integrado ao SDK do servidor versão 5.3.x ou posterior, você pode pular esta etapa.

Para servidores de jogos integrados ao SDK do servidor versão 5.2 ou anterior, você precisa localizar e copiar manualmente as bibliotecas. Você deve usar a mesma versão do OpenSSL que sua versão do Unreal Engine 5 usa. As compilações de jogos implantadas com as bibliotecas OpenSSL erradas não conseguirão se comunicar com o serviço. Amazon GameLift Servers

- a. Procure as bibliotecas do OpenSSL na fonte do seu mecanismo de jogo. A localização varia de acordo com seu ambiente de desenvolvimento:

No 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

No 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. Copie as bibliotecas do OpenSSL para o diretório do pacote de compilação do jogo em. <YourGame>/Binaries/Win64

Para preparar sua compilação de servidor para implantação na nuvem (Linux)

Para obter instruções mais detalhadas sobre como preparar um servidor de jogos criado para Linux, consulte [Criação do SDK do servidor Amazon GameLift Servers para o Unreal Engine 5 no Amazon Linux](#).

1. Designe um diretório de trabalho para organizar seus arquivos de compilação. A estrutura do diretório de trabalho é implantada como está em cada computação de hospedagem. Adicione seu servidor de jogos construído em Linux e todos os arquivos dependentes.
2. Crie um script de instalação de compilação do servidor na raiz do seu diretório de trabalho. Se necessário, crie um `install.sh` arquivo e adicione os comandos necessários para instalar corretamente a compilação do servidor do jogo. Amazon GameLift Servers usa esse arquivo para instalar o servidor criado em cada recurso de EC2 hospedagem.
3. Adicione os arquivos da biblioteca OpenSSL à compilação do seu servidor de jogos.

 Note

Se o seu servidor de jogos estiver integrado ao SDK do servidor versão 5.3.x ou posterior, você pode pular esta etapa.

Para servidores de jogos integrados ao SDK do servidor versão 5.2 ou anterior, você precisa localizar e copiar manualmente as bibliotecas. Você deve usar a mesma versão do OpenSSL que sua versão do Unreal Engine 5 usa. As compilações de jogos implantadas com as bibliotecas OpenSSL erradas não conseguirão se comunicar com o serviço. Amazon GameLift Servers

- a. Procure as bibliotecas do OpenSSL na fonte do seu mecanismo de jogo. A localização varia de acordo com seu ambiente de desenvolvimento:

No 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`

No 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. Ao localizar as bibliotecas do OpenSSL, copie-as para o diretório do pacote de compilação do jogo em. `<YourGame>/Binaries/Linux`

Etapa 3: selecionar o cenário de implantação

Nesta etapa, você escolhe a solução de hospedagem de jogos que deseja implantar no momento. Você poderá ter várias implantações do seu jogo, usando qualquer um dos cenários.

- **Frota de região única:** implanta seu servidor de jogo em uma única frota de recursos de hospedagem na região padrão AWS do perfil ativo. Esse cenário é um bom ponto de partida para testar a integração do servidor da AWS e a configuração de compilação do servidor. Ele implanta os seguintes recursos:
 - frota (sob demanda) da AWS com a versão do seu servidor de jogos instalada e em execução.
 - Grupo de usuários e cliente do Amazon Cognito para permitir que os jogadores se autentiquem e iniciem um jogo.
 - Autorizador de gateway de API que vincula o grupo de usuários a APIs
 - Web ACL para limitar chamadas excessivas de jogadores para o gateway da API.
 - Gateway de API + função do Lambda para os jogadores solicitarem um slot de jogo. Essa função é chamada `CreateGameSession()` se nenhuma estiver disponível.
 - Gateway de API + função do Lambda para que os jogadores obtenham informações de conexão para sua solicitação de jogo.
- **FlexMatch frota:** implanta seu servidor de jogo em um conjunto de frotas e configura um FlexMatch matchmaker com regras para criar partidas de jogadores. Este cenário usa hospedagem spot de baixo custo com uma estrutura de várias frotas e vários locais para uma disponibilidade duradoura. Esta abordagem é útil quando se está pronto para começar a projetar um componente de matchmaker para a solução de hospedagem. Nesse cenário, você criará os recursos básicos para essa solução, que poderão ser personalizados posteriormente, conforme necessário. Ele implanta os seguintes recursos:
 - FlexMatch configuração de matchmaking e regras de matchmaking definidas para aceitar solicitações de jogadores e formar partidas.
 - Três frotas da AWS com sua versão de servidor de jogo instalada e funcionando em vários locais. Inclui duas frotas spot e uma frota sob demanda como backup.
 - Fila de posicionamento de sessões de jogo da AWS que atende às solicitações de partidas propostas, encontrando o melhor recurso de hospedagem possível (com base na viabilidade, custo, latência do jogador etc.) e iniciando uma sessão de jogo.
 - Grupo de usuários e cliente do Amazon Cognito para permitir que os jogadores se autentiquem e iniciem um jogo.
 - Autorizador de gateway de API que vincula o grupo de usuários a APIs

- Web API para limitar chamadas excessivas de jogadores para o gateway da API.
- Gateway de API + função do Lambda para os jogadores solicitarem um slot de jogo. Esta função chama `StartMatchmaking()`.
- Gateway de API + função do Lambda para que os jogadores obtenham informações de conexão para sua solicitação de jogo.
- Tabelas do Amazon DynamoDB para armazenar ingressos de criação de parcerias para jogadores e informações da sessão do jogo.
- Tópico do SNS + Função Lambda para `GameSessionQueue` lidar com eventos.

Etapa 4: configurar parâmetros do jogo

Nesta etapa, você descreve seu jogo para ser carregado em; AWS

- Nome da compilação do servidor: forneça um nome significativo para a compilação do servidor de jogos. A AWS usa esse nome para se referir à cópia da compilação do servidor, que é carregada e usada para implantações.
- Sistema operacional de criação de servidor: insira o sistema operacional no qual o servidor foi criado para ser executado. Isso informa a AWS que tipo de recursos computacionais usar para hospedar seu jogo.
- Pasta do servidor do jogo: identifique o caminho para a pasta de criação do servidor local.
- Compilação do servidor de jogo: identifique o caminho para o executável do servidor de jogos.
- Caminho do cliente do jogo: identifique o caminho para o executável do cliente do jogo.
- Saída de configuração do cliente: esse campo precisa apontar para uma pasta na compilação do cliente que contém sua AWS configuração. Procure-o no seguinte local: `[client-build]/[project-name]/Content/CloudFormation`.

Etapa 5: implantar o cenário

Nesta etapa, você implanta seu jogo em uma solução de hospedagem em nuvem com base no cenário de implantação escolhido. Esse processo pode levar vários minutos enquanto a AWS valida a compilação do servidor, provisiona recursos de hospedagem, instala o servidor de jogos, inicia os processos do servidor e os prepara para hospedar sessões de jogo.

Para iniciar a implantação, escolha **Implantar CloudFormation**. É possível acompanhar o status de sua hospedagem de jogos aqui. Para obter informações mais detalhadas, você pode entrar no

console AWS de gerenciamento AWS e visualizar notificações de eventos. Certifique-se de fazer login usando a mesma conta, usuário e AWS região do perfil de usuário ativo no plug-in.

Quando a implantação estiver concluída, você terá seu servidor de jogo instalado em uma AWS EC2 instância. Pelo menos um processo do servidor está em execução e pronto para iniciar uma sessão de jogo.

Etapa 6: Inicializar o cliente

Neste ponto, você concluiu todas as tarefas necessárias para iniciar e jogar seu jogo multijogador hospedado com Amazon GameLift Servers. Para jogar, inicie uma instância do seu cliente de jogo.

Se você implantou o cenário de frota única, poderá abrir uma única instância de cliente com um jogador, entrar no mapa do servidor e se movimentar. Abra instâncias adicionais do cliente do jogo para adicionar um segundo jogador ao mesmo mapa de jogo do servidor.

Se você implantou o FlexMatch cenário, a solução espera que pelo menos dois clientes sejam colocados na fila para a colocação da sessão de jogo antes que os jogadores possam entrar no mapa do servidor.

Plugin para Unreal: implante seu jogo em uma frota de contêineres gerenciada

Use esse fluxo de trabalho de plug-in guiado para criar uma imagem de contêiner para seu servidor de jogos e implantá-la em uma solução de hospedagem baseada em contêiner. Se você ainda não integrou o código do jogo, consulte [Plugin para Unreal: integre seu código de jogo](#). Quando você concluir esse fluxo de trabalho com êxito, seu servidor de jogos em contêiner estará sendo executado na nuvem e você poderá usar o plug-in para iniciar um cliente de jogo, conectar-se a uma sessão de jogo e jogar o jogo.

Antes de começar

Esse fluxo de trabalho pressupõe que você tenha concluído as seguintes tarefas.

- Integre o código do servidor do jogo com o SDK Amazon GameLift Servers do servidor. Seu servidor de jogo hospedado deve ser capaz de se comunicar com o Amazon GameLift Servers serviço para que ele possa responder às solicitações para iniciar novas sessões de jogo e relatar o status da sessão do jogo. Se você não concluiu essa tarefa, recomendamos que você siga primeiro o fluxo de trabalho do plug-in Host with Anywhere. Para obter orientação sobre como preparar o código do servidor do jogo, consulte [Atualize o código do seu servidor de jogos](#). Para uma frota de contêineres gerenciada, você deve integrar seu jogo com o SDK do servidor versão 5.2 ou superior.

Note

Se você estiver usando o mapa do jogo de inicialização, essa tarefa já foi feita para você.

- Package o executável do seu servidor de jogos para rodar no Linux. [Se você estiver desenvolvendo no Windows e integrando o SDK do servidor C++ versão 5.2.x ou anterior, precisará trabalhar com o kit de ferramentas de compilação cruzada Unreal](#). Como alternativa, você pode configurar um espaço de trabalho Linux separado ou usar uma ferramenta como o subsistema Windows para Linux (WSL).
- Reúna arquivos para implantar com a criação do seu servidor de jogos. Em sua máquina local, crie um diretório de trabalho para organizar os arquivos, que serão incorporados à imagem do contêiner do seu servidor de jogos. Isso pode incluir dependências do jogo, um script para iniciar servidores de jogos e outros processos ao iniciar um contêiner, etc.
- Adicione os arquivos da biblioteca OpenSSL para a construção do seu servidor de jogos.

Note

Se o seu servidor de jogos estiver integrado ao SDK do servidor versão 5.3.x ou posterior, você pode pular esta etapa.

Para servidores de jogos integrados ao SDK do servidor versão 5.2 ou anterior, você precisa localizar e copiar manualmente as bibliotecas. Você deve usar a mesma versão do OpenSSL que sua versão do Unreal Engine 5 usa. As compilações de jogos implantadas com as bibliotecas OpenSSL erradas não conseguirão se comunicar com o serviço. Amazon GameLift Servers

- a. Procure as bibliotecas do OpenSSL na fonte do seu mecanismo de jogo. A localização varia de acordo com seu ambiente de desenvolvimento:

No 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

No 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. Copie as bibliotecas do OpenSSL para o diretório do pacote de compilação do jogo em.
<YourGame>/Binaries/Win64
- Integre seu código de cliente de jogo com Amazon GameLift Servers o. Uma forma de concluir essa tarefa é adicionar um ativo de amostra (incluído no plug-in) que já esteja integrado. Para obter orientação sobre como preparar o código do cliente do jogo, consulte [Integre o mapa de jogo do seu cliente](#).
- Instale o Docker em sua máquina local. Você precisa instalar essa ferramenta se quiser que o plug-in crie imagens de contêiner para você e as envie para um repositório ECR. Como alternativa, você pode realizar essas tarefas manualmente e instruir o plug-in a usar uma imagem de contêiner existente. Para obter mais informações sobre como criar sua imagem manualmente, consulte [Criar uma imagem de contêiner para Amazon GameLift Servers](#).

Para iniciar o fluxo de trabalho de contêineres Amazon GameLift Servers gerenciados:

- Na barra de ferramentas principal do editor Unreal, escolha o Amazon GameLift Servers menu e selecione Contêineres gerenciados. Essa ação abre a página do plug-in Host with Managed Containers, que apresenta um step-by-step processo para criar uma imagem de contêiner com a versão do seu servidor de jogos, implantá-la em uma frota de contêineres e iniciar seu jogo.

Etapa 0: defina seu perfil

Esta seção exibe seu perfil de usuário atualmente selecionado. Verifique se o perfil de usuário atual é o que você deseja usar para esse fluxo de trabalho. Todos os recursos que você cria nesse fluxo de trabalho são associados à AWS conta do perfil e colocados na AWS região padrão do perfil. As permissões do usuário do perfil determinam seu acesso aos AWS recursos e ações.

Talvez seja necessário modificar o perfil de usuário selecionado se:

- Nenhum perfil está selecionado no momento.
- Você deseja selecionar um perfil diferente ou criar um novo perfil.
- Você precisa inicializar o perfil selecionado (se o status do bootstrap estiver inativo).

Para definir ou alterar o perfil de usuário selecionado

- No Amazon GameLift Servers menu, escolha Credenciais de AWS acesso aberto.

Etapa 1: avaliar a prontidão do contêiner

Antes de implantar seu servidor de jogo em uma frota de contêineres, você deve empacotá-lo em uma imagem de contêiner e armazená-lo em um repositório Amazon ECR. O plug-in pode concluir essas tarefas para você ou você pode fazer essas tarefas manualmente. Nesta etapa, forneça informações sobre o status da imagem do contêiner e do repositório ECR.

Use as perguntas de avaliação para dizer ao plug-in quais etapas ele precisa seguir:

- Crie uma nova imagem de contêiner. Se você escolher essa opção, a próxima etapa solicitará a localização do diretório de compilação do servidor do jogo e do executável de compilação. O plug-in usa um modelo Dockerfile (fornecido por Amazon GameLift Servers) e o configura automaticamente para o seu jogo. Você pode ver o modelo em [Crie uma imagem de contêiner para Amazon GameLift Servers](#). Depois de escolher essa opção, indique onde você deseja que o plugin armazene a nova imagem:
 - Crie um novo repositório Amazon ECR e envie a imagem do contêiner para ele. O plug-in cria um repositório ECR privado usando a AWS conta e o padrão Região da AWS no perfil de usuário selecionado.
 - Envie a imagem do contêiner para um repositório Amazon ECR criado anteriormente. Se você escolher essa opção, a próxima etapa solicitará que você selecione um repositório Amazon ECR existente em uma lista. A lista inclui todos os repositórios Amazon ECR da AWS conta e do padrão Região da AWS no perfil de usuário selecionado. Você pode selecionar um repositório público ou privado.
- Use uma imagem de contêiner existente. Se você criou uma imagem manualmente, recomendamos que você use o modelo Dockerfile fornecido pela Amazon GameLift Servers, que está disponível em. [Crie uma imagem de contêiner para Amazon GameLift Servers](#) Depois de escolher essa opção, indique onde a imagem está localizada.
 - Uma imagem gerada pelo Docker armazenada localmente. Se você escolher essa opção, o plug-in cria um novo repositório privado do Amazon ECR e envia o arquivo de imagem local para ele. A próxima etapa solicitará um ID de imagem, que o plug-in usa para localizar o arquivo de imagem.
 - Uma imagem de contêiner que já está armazenada em um repositório Amazon ECR. Se você escolher essa opção, a próxima etapa solicitará que você selecione um repositório e uma

imagem existentes do Amazon ECR em uma lista. A lista inclui todos os repositórios Amazon ECR da AWS conta e do padrão Região da AWS no perfil de usuário selecionado. Você pode selecionar um repositório público ou privado.

Etapa 2: Configurar a implantação da imagem

Nesta etapa, forneça as informações de que o plug-in precisa para implantar sua imagem de contêiner em uma frota de contêineres. Essa etapa solicita as seguintes informações:

- A localização da compilação do seu servidor de jogos, da imagem do contêiner ou do repositório Amazon ECR, com base nas suas seleções na Etapa 1.
- O cenário a ser usado para a implantação de seus contêineres gerenciados.
- O caminho de saída da configuração do cliente. Selecione a pasta na compilação do cliente que contém sua AWS configuração. Procure-o no seguinte local: `[client-build]/[project-name]/Content/CloudFormation`.
- Configurações de implantação opcionais. Esta seção tem configurações que o plug-in usa por padrão. Você pode modificá-los ou manter os valores padrão.
 - O nome do jogo é definido como o nome do seu projeto de jogo por padrão. Todos os AWS recursos criados pelo plug-in fazem referência ao valor do nome do jogo.
 - O intervalo de portas, o limite de memória e o limite de vCPU são configurações para a frota de contêineres. Para obter mais informações sobre a personalização desses valores, consulte o intervalo [Configurar conexões de rede](#) de portas de conexão e os limites [Defina limites de recursos](#) de recursos.
 - A tag de imagem de contêiner é usada para categorizar suas imagens de contêiner no Amazon ECR. O valor padrão é `unreal-gamelift-plugin`.
 - Nome do repositório Amazon ECR. Você pode editar esse campo para sugerir um nome personalizado somente quando o plug-in estiver criando um repositório ECR para você. O valor padrão é `unreal-game lift-plugin-ecr-repository`.

Opções de cenário de implantação

Frota de contêineres de região única

Esse cenário implanta seu servidor de jogo em uma única frota de contêineres. É um bom ponto de partida para testar a integração do servidor AWS e a configuração do contêiner. Ele implanta os seguintes recursos.

- Amazon GameLift ServersA definição de grupo de contêineres descreve como implantar e executar suas imagens de contêineres em uma frota de contêineres.
- Amazon GameLift Serversfrota de contêineres (sob demanda) com seu contêiner de servidor de jogos instalado e em execução, com alias.
- Grupo de usuários e cliente do Amazon Cognito para permitir que os jogadores se autentiquem e iniciem um jogo.
- Autorizador do API Gateway que vincula o grupo de usuários a APIs
- Lista de controle de acesso à Web (ACL) para limitar chamadas excessivas de jogadores para o API Gateway.
- Serviço de back-end para fazer solicitações ao Amazon GameLift Servers serviço em nome de clientes de jogos, como solicitar sessões de jogos e participar de jogos:
 - API Gateway + função Lambda para os jogadores solicitarem um slot para uma sessão de jogo. Essa função é chamada `CreateGameSession()` se nenhum slot aberto estiver disponível.
 - API Gateway + função Lambda para que os jogadores obtenham informações de conexão para sua solicitação de jogo.

Frota de contêineres de uma única região com FlexMatch

Esse cenário implanta seu servidor de jogo em uma frota de contêineres, configura o posicionamento da sessão de jogo e FlexMatch configura o matchmaking. Esse cenário é útil quando você está pronto para começar a criar um matchmaker personalizado para sua solução de hospedagem. Use esse cenário para criar os recursos básicos para essa solução, que você pode personalizar posteriormente, conforme necessário. Ele implanta os seguintes recursos:

- Amazon GameLift Serversdefinição de grupo de contêineres que descreve como implantar e executar suas imagens de contêineres em uma frota de contêineres.
- Amazon GameLift Serversfrota de contêineres (sob demanda) com seu contêiner de servidor de jogos instalado e em execução, com alias.
- FlexMatchconfiguração de matchmaking e regras de matchmaking definidas para aceitar solicitações de jogadores e formar partidas.
- Amazon GameLift Serversfila de sessões de jogo que atende às solicitações de partidas propostas, encontrando o melhor recurso de hospedagem possível (com base na viabilidade, custo, latência do jogador etc.) e iniciando uma sessão de jogo.
- Grupo de usuários e cliente do Amazon Cognito para permitir que os jogadores se autentiquem e iniciem um jogo.

- Autorizador do API Gateway que vincula o grupo de usuários a APIs
- Lista de controle de acesso à Web (ACL) para limitar chamadas excessivas de jogadores para o API Gateway.
- Serviço de back-end para fazer solicitações ao Amazon GameLift Servers serviço em nome de clientes de jogos, como solicitar sessões de jogos e participar de jogos:
 - API Gateway + função Lambda para os jogadores solicitarem um slot para uma sessão de jogo. Essa função é chamada `StartMatchmaking()` se nenhum slot aberto estiver disponível.
 - API Gateway + função Lambda para que os jogadores obtenham informações de conexão para sua solicitação de jogo.
- Tabelas do DynamoDB para armazenar tíquetes de matchmaking para jogadores e informações da sessão do jogo.
- Tópico do Amazon SNS + função Lambda para lidar com eventos. `GameSessionQueue`

Implantar frota de contêineres

Quando a configuração da frota estiver concluída, escolha o botão **Implantar frota de contêineres** para iniciar a implantação. Esse processo pode levar vários minutos enquanto o plug-in cria uma imagem de contêiner e a envia para o ECR, provisiona recursos de hospedagem para a frota de contêineres, implanta a frota e outros AWS recursos para o cenário de solução de hospedagem selecionado.

Ao iniciar uma implantação, você pode acompanhar o progresso de cada etapa. Dependendo da sua configuração, as etapas podem incluir o seguinte:

- Configurando a imagem do contêiner
- Criação de um repositório do Amazon ECR
- Criando uma imagem e enviando para o Amazon ECR
- Criação de definição de grupo de contêiner
- Criando frota de contêineres

Para obter informações de implantação mais detalhadas, escolha **Exibir** no console AWS de gerenciamento. Quando a frota de contêineres atinge o status ativo, a frota está executando ativamente contêineres com processos de servidor prontos para hospedar sessões de jogo.

Quando a implantação estiver concluída, você terá uma frota de contêineres em funcionamento pronta para hospedar sessões de jogo e aceitar conexões de jogadores.

Você não pode interromper uma implantação em andamento. Se a implantação entrar em um estado ruim ou falhar, você poderá recomeçar usando a opção Redefinir implantação.

Cliente de lançamento

Neste ponto, você concluiu todas as tarefas para iniciar e jogar seu jogo multijogador hospedado Amazon GameLift Servers. Para jogar seu jogo, escolha Start Client para iniciar uma instância local do seu cliente de jogo.

- Se você implantou o cenário de frota única, abra uma instância do seu cliente de jogo com um jogador e entre no mapa do servidor para se movimentar. Você pode abrir uma segunda instância do cliente do jogo para adicionar um segundo jogador ao mesmo mapa de jogo do servidor.
- Se você implantou o FlexMatch cenário, a solução de hospedagem espera que pelo menos dois clientes do jogo façam solicitações de matchmaking. Abra pelo menos duas instâncias do seu cliente de jogo com um jogador. Os dois jogadores serão emparelhados e solicitados a participar de uma sessão de jogo para a partida.

Atualize uma frota de contêineres

Se você implantou com sucesso uma solução de hospedagem de contêineres gerenciados, poderá usar o recurso Atualizar implantação. Essa opção permite que você atualize as configurações de uma frota de contêineres implantada, sem precisar criar uma nova frota.

Ao atualizar uma implantação, você pode implantar uma imagem de contêiner com uma versão de servidor de jogo diferente, alterar o repositório do Amazon ECR, escolher um cenário de implantação diferente e personalizar as configurações opcionais.

Quando você estiver pronto para implantar suas alterações, escolha Atualizar. O tempo necessário para uma atualização de implantação é semelhante ao de uma implantação completa. Para obter informações detalhadas de implantação, escolha Exibir no console AWS de gerenciamento.

Limpe os recursos implantados

Como prática recomendada, limpe os AWS recursos de sua solução de contêineres gerenciados assim que você não precisar mais deles. Você pode continuar incorrendo em custos com esses recursos se não os remover.

Exclua os seguintes recursos:

- Pilha gerenciada de recursos de contêineres. Os recursos nessa pilha dependem do cenário de implantação selecionado. Para excluir a pilha inteira, use o AWS CloudFormation console. As pilhas geradas a partir do Amazon GameLift Servers plug-in usam a seguinte convenção de nomenclatura: `GameLiftPluginForUnreal-{GameName}-Containers`. Aguarde a conclusão do processo de exclusão da pilha antes de iniciar uma nova implantação de contêineres gerenciados no plug-in. Para obter mais informações, consulte [Excluir uma pilha do CloudFormation console](#).
- Repositório Amazon ECR. Se você usou o plug-in para criar um repositório para sua imagem de contêiner, talvez queira excluir os repositórios que não são mais necessários. Você não precisa excluir um repositório antes de redefinir uma implantação de contêineres gerenciados. Se você atualizar ou redefinir uma implantação, o plug-in usará automaticamente o mesmo repositório, a menos que seja orientado a usar outro. Para obter mais informações, consulte [Excluir um repositório privado no Amazon ECR](#).

Amazon GameLift Serversplugin para Unity (servidor SDK 5.x)

Esse plug-in adiciona o SDK e as ferramentas do servidor Amazon GameLift Servers C# ao editor UE. Use os fluxos de trabalho guiados da interface do usuário para integrar a funcionalidade do SDK do servidor em seu projeto de jogo e implantar uma solução de Amazon GameLift Servers hospedagem para seu servidor de jogos.

Com o plug-in, você pode criar uma solução básica de hospedagem funcional e, em seguida, otimizar e personalizar conforme necessário. Configure uma frota Amazon GameLift Servers Anywhere com sua estação de trabalho local como anfitriã. Para hospedagem na nuvem com frotas de contêineres gerenciadas EC2 ou gerenciadas, implante seu servidor de jogos com uma solução completa para gerenciar solicitações de sessões de jogos e conexões de clientes.

Tópicos

- [Instale o plug-in para seu projeto de jogo Unreal](#)
- [Plugin para Unity: configure um perfil de AWS usuário](#)
- [Plugin para Unity: configure testes locais com Amazon GameLift Servers Anywhere](#)
- [Plugin para Unity: implante seu jogo em uma EC2 frota gerenciada](#)
- [Plugin para Unity: implante seu jogo em uma frota de contêineres gerenciada](#)

Instale o plug-in para seu projeto de jogo Unreal

[Obtenha o Amazon GameLift Servers plug-in para Unity em GitHub](#)

Consulte o readme do GitHub repositório para obter informações sobre como instalar o plug-in em um projeto de jogo.

O plug-in inclui os seguintes componentes:

- Módulos de plug-in para o editor do Unity. Quando o plug-in é instalado, um novo item do menu principal fornece acesso à Amazon GameLift Servers funcionalidade.
- Bibliotecas C# para a API Amazon GameLift Servers de serviço com funcionalidade do lado do cliente.
- Bibliotecas C# para o SDK Amazon GameLift Servers do servidor (versão 5.x).
- Experimente o conteúdo do jogo, incluindo recursos e cenas, para que você possa experimentar Amazon GameLift Servers mesmo que não tenha um jogo multijogador pronto para ser construído.
- Configurações da solução, fornecidas como AWS CloudFormation modelos, que o plug-in usa ao implantar seu servidor de jogos na nuvem para hospedagem.

Esse plug-in usa modelos AWS CloudFormation para implantar soluções de hospedagem para cenários comuns de jogos. Use essas soluções conforme fornecidas ou personalize-as conforme necessário para seus jogos.

Plugin para Unity: configure um perfil de AWS usuário

Depois de instalar o plug-in, configure um perfil de usuário com uma AWS conta válida. Você pode manter vários perfis no plug-in, mas só pode selecionar um perfil por vez. Sempre que você trabalha no plug-in, selecione um perfil para usar. Cada página do fluxo de trabalho exibe o perfil atualmente selecionado.

A manutenção de vários perfis permite alternar entre diferentes implantações de hospedagem. Por exemplo, você pode configurar perfis que usam a mesma AWS conta, mas implantados em AWS regiões diferentes. Ou você pode configurar perfis com AWS contas ou usuários e conjuntos de permissões diferentes.

Note

Se você instalou a AWS CLI em sua estação de trabalho e já tem um perfil configurado, o Amazon GameLift Servers plug-in o detectará e o listará como um perfil existente. O plug-in

seleciona automaticamente qualquer perfil nomeado [default]. Você poderá usar um perfil existente ou criar outro.

Todos os perfis devem ser inicializados para configurar alguns AWS recursos necessários em sua conta de usuário.

Para gerenciar seus AWS perfis

1. Na barra de ferramentas principal do Unity, escolha o Amazon GameLift Servers menu e selecione Credenciais de AWS acesso. Essa ação abre o Amazon GameLift Servers plug-in na página Configurar seu perfil de usuário.
2. Use os botões para criar uma nova AWS conta ou configurar um perfil de usuário para uma AWS conta que você já tem.
3. Se você ainda não tiver um perfil de usuário, será solicitado que você insira os detalhes do perfil e crie um novo perfil. Forneça as informações a seguir:
 - Uma AWS conta. Se você criar uma nova AWS conta, use o link para o AWS Management Console e siga as instruções. Consulte [Criar uma conta da AWS](#) para obter mais detalhes.
 - Um AWS usuário com permissões de uso Amazon GameLift Servers e outros AWS serviços necessários. Consulte [Configurar um Conta da AWS](#) para obter instruções sobre como configurar um usuário AWS Identity and Access Management (IAM) com Amazon GameLift Servers permissões e acesso programático com credenciais de longo prazo.
 - Credenciais para seu AWS usuário. Essas credenciais consistem em um ID de chave de AWS acesso e uma chave AWS secreta. Consulte [Obter suas chaves de acesso](#) para obter mais detalhes.
 - AWS região. Essa é uma localização geográfica em que você deseja criar seus AWS recursos para hospedagem. Durante o desenvolvimento, recomendamos usar uma região próxima à sua localização física. Escolha uma região na lista de [AWS regiões suportadas](#).
4. Se o plug-in detectar perfis existentes, ele exibirá uma lista dos perfis disponíveis. Selecione um perfil existente na lista ou escolha Adicionar outro perfil para criar um novo.

Inicialize um perfil de usuário

Todos os perfis devem ser inicializados para serem usados com o plug-in. Amazon GameLift Servers O Bootstrapping cria um bucket do Amazon S3 específico para o perfil. Ele é usado para

armazenar configurações de projetos, compilar artefatos e outras dependências. Os buckets não são compartilhados entre outros perfis.

O bootstrapping envolve a criação de novos AWS recursos e pode gerar custos.

Para inicializar seu perfil:

1. Na página Credenciais de AWS acesso, verifique o status de bootstrap do perfil de usuário que você deseja usar. Se o status de bootstrap do perfil for "Inativo" e não houver nenhum bucket do S3 listado, você precisará inicializar o perfil.
2. Selecione o perfil que você deseja usar e escolha o perfil Bootstrap.
3. Aguarde até que o status do bootstrap mude para "Active". Isso pode levar alguns minutos.

Plugin para Unity: configure testes locais com Amazon GameLift Servers Anywhere

Nesse fluxo de trabalho, você adiciona código de jogo de cliente e servidor para Amazon GameLift Servers funcionalidade e usa o plug-in para designar sua estação de trabalho local como host de servidor de jogos de teste. Depois de concluir as tarefas de integração, use o plug-in para criar seus componentes de cliente e servidor do jogo.

Para iniciar o fluxo de trabalho Amazon GameLift Servers Anywhere:

- No menu principal do editor Unity, escolha Amazon GameLift Server e selecione Host with Anywhere. Essa ação abre a página do plugin para configurar seu jogo com uma frota @Anywhere. A página apresenta um processo de cinco etapas para integrar, compilar e iniciar os componentes do jogo.

Definir seu perfil

Escolha o perfil que você deseja usar ao seguir esse fluxo de trabalho. O perfil selecionado afeta todas as etapas do fluxo de trabalho. Todos os recursos que você cria estão associados à AWS conta do perfil e são colocados na AWS região padrão do perfil. As permissões do usuário do perfil determinam seu acesso aos AWS recursos e ações.

1. Selecione um perfil na lista suspensa de perfis disponíveis. Se você ainda não tem um perfil ou deseja criar um novo, acesse o Amazon GameLift Servers menu e escolha Definir perfis de AWS conta.

2. Se o status do bootstrap não for “Ativo”, escolha o perfil do Bootstrap e espere que o status mude para “Ativo”.

Integre seu código de jogo com o SDK do servidor C#

 Note

Se você importou o jogo de exemplo, poderá ignorar esta etapa. Os ativos do jogo de exemplo já têm o código necessário para o servidor e o cliente.

Nessa etapa do fluxo de trabalho, você atualizará o código do cliente e do servidor em seu projeto de jogo.

- * Os servidores de jogos devem ser capazes de se comunicar com o Amazon GameLift Servers serviço para receber solicitações para iniciar uma sessão de jogo, fornecer informações de conexão da sessão de jogo e relatar o status.
- Os clientes do jogo devem obter informações sobre sessões de jogo, entrar ou iniciar sessões e obter informações de conexão para entrar em um jogo.

Integrar o código do servidor

Se estiver usando seu próprio projeto de jogo com cenas personalizadas, use o código de exemplo fornecido para adicionar o código de servidor necessário ao seu projeto de jogo:

1. Nos arquivos do seu projeto de jogo, abra a pasta `Assets/Scripts/Server`. Se não existir, crie-a.
2. Vá para o GitHub repositório [aws/ amazon-gamelift-plugin-unity](https://github.com/aws/amazon-gamelift-plugin-unity) e abra o caminho. `Samples~/SampleGame/Assets/Scripts/Server`
3. Localize o arquivo `GameLiftServer.cs` e copie-o para a pasta `Server` do projeto de jogo. Ao compilar um executável de servidor, use esse arquivo como destino da compilação.

O código de exemplo inclui esses elementos mínimos necessários, que usam o SDK do servidor Amazon GameLift Servers C# (versão 5):

- Inicializa um cliente de Amazon GameLift Servers API. A `InitSDK()` chamada com os parâmetros do servidor é necessária para uma frota Amazon GameLift Servers Anywhere. Essas configurações são definidas automaticamente para uso no plug-in.
- Implementa as funções de retorno de chamada necessárias para responder às solicitações do Amazon GameLift Servers serviço, incluindo `OnStartGameSessionOnProcessTerminate`, e, `onHealthCheck`
- Chamadas `ProcessReady()` com uma porta designada para notificar o Amazon GameLift Servers serviço quando o processo do servidor estiver pronto para hospedar sessões de jogo.

Se quiser personalizar o código do servidor de exemplo, consulte estes recursos:

- [Adicione Amazon GameLift Servers ao seu servidor de jogos](#)
- [SDK 5.x do servidor C# para Amazon GameLift Servers -- Ações](#)

Integrar o código do cliente

Se estiver usando seu próprio projeto de jogo com cenas personalizadas, você precisará integrar a funcionalidade básica ao cliente do jogo. Também será necessário adicionar elementos de interface do usuário para que os jogadores possam entrar e participar de uma sessão de jogo. Use a API de serviço para Amazon GameLift Servers (no AWS SDK) obter informações da sessão do jogo, criar novas sessões de jogo ou participar de sessões de jogos existentes,

Ao criar um cliente para testes locais com uma frota Anywhere, você pode adicionar chamadas diretas ao Amazon GameLift Servers serviço. Ao desenvolver seu jogo para hospedagem na nuvem, ou se planeja usar frotas Anywhere para hospedagem de produção, você precisará criar um serviço de back-end do lado do cliente para lidar com toda a comunicação entre os clientes do jogo e o serviço. Amazon GameLift Servers

Para Amazon GameLift Servers integrar seu código de cliente, use os seguintes recursos como guia.

- Integre o cliente com a `GameLiftCoreApi` classe no GitHub repositório `amazon-gamelift-plugin-unity aws/`. Essa classe fornece controles para autenticação do jogador e para recuperar informações de sessões de jogo.
- Veja exemplos de integrações de jogos, disponíveis no GitHub repositório `amazon-gamelift-plugin-unity aws/`,. `Samples~/SampleGame/Assets/Scripts/Client/GameLiftClient.cs`
- Siga as instruções em Adicionar Amazon GameLift Servers ao seu cliente de jogo Unity.

Para clientes do jogo que se conectam a uma frota Anywhere, os clientes de jogos precisam das informações a seguir. O plug-in atualiza automaticamente o projeto de jogo para usar os recursos que você criou no plug-in.

- **FleetId** - O identificador exclusivo da sua frota Anywhere.
- **FleetLocation** - A localização personalizada da sua frota Anywhere.
- **AwsRegion** - A AWS região onde sua frota Anywhere está hospedada. Essa é a região que você definiu no seu perfil de usuário.
- **ProfileName** - Um perfil de AWS credenciais em sua máquina local que permite acesso ao AWS SDK para Amazon GameLift Servers. O cliente do jogo usa essas credenciais para autenticar solicitações ao Amazon GameLift Servers serviço.

 Note

O perfil de credenciais é gerado pelo plug-in e armazenado na máquina local. Consequentemente, é necessário executar o cliente na máquina local (ou em uma máquina com o mesmo perfil).

Conectar-se a uma frota Anywhere

Nesta etapa, você designa uma frota do Anywhere para usar. Uma frota do Anywhere define uma coleção de recursos computacionais, que podem estar localizados em qualquer lugar, para hospedagem de servidores de jogos.

- Se a AWS conta que você está usando atualmente tiver frotas Anywhere existentes, abra o campo suspenso Nome da frota e escolha uma frota. Essa lista suspensa mostra apenas as frotas de Anywhere na AWS região para o perfil de usuário atualmente ativo.
- Se não houver frotas existentes, ou se você quiser criar uma, escolha Criar frota Anywhere e forneça um nome para a frota.

Depois de escolher uma frota Anywhere para seu projeto, Amazon GameLift Servers verifique se o status da frota está ativo e exibe o ID da frota. Você poderá acompanhar o progresso dessa solicitação no log de saída do editor do Unity.

Registrar um computador

Nesta etapa, você registra sua estação de trabalho local como um recurso computacional na nova frota do Anywhere.

1. Insira um nome de computação para a sua máquina local. Se você adicionar mais de uma computação na frota, os nomes deverão ser exclusivos.
2. Escolha Registrar computação. Você poderá acompanhar o progresso dessa solicitação no log de saída do editor do Unity.

O plug-in registra a estação de trabalho local com o endereço IP definido como localhost (127.0.0.1). Esta configuração pressupõe que você executará o cliente e o servidor de jogo na mesma máquina.

Em resposta a essa ação, Amazon GameLift Servers verifica se ela pode se conectar à computação e retorna informações sobre a computação recém-registrada.

Iniciar o jogo

Nesta etapa, você compila os componentes do jogo e os inicia para jogar. Complete as seguintes tarefas:

1. Configure o cliente do jogo. Nesta etapa, você solicita que o plug-in atualize um ativo `GameLiftClientSettings` para o projeto de jogo. O plug-in usa esse recurso para armazenar determinadas informações que seu cliente de jogo precisa para se conectar ao Amazon GameLift Servers serviço.
 - a. Se você não importou nem inicializou o jogo de exemplo, crie um ativo `GameLiftClientSettings`. No menu principal do editor Unity, escolha Assets, Create, Amazon GameLift, Client Settings. Se você criar várias cópias `GameLiftClientSettings` em seu projeto, o plug-in detectará isso automaticamente e notificará qual ativo o plug-in atualizará.
 - b. Em Iniciar o jogo, escolha Configurar cliente: aplicar configurações do Anywhere. Essa ação atualiza as configurações do cliente do jogo para usar a frota Anywhere que você acabou de configurar.
2. Compile e execute o cliente do jogo.
 - a. Compile um executável de cliente usando o processo de compilação padrão do Unity. Em Arquivo, Configurações de compilação, altere a plataforma para Windows, Mac, Linux. Se

you imported the example game and initialized the configurations, the list of compilations and the destination of the compilation will be updated automatically.

- b. Start one or more instances of the recently-created game client.
3. Start a game server in the Anywhere fleet. Choose **Server: start the server in the editor**. This task starts an active server to which the client can connect, as long as the Unity editor remains open.
4. Start or participate in a game session. In the game client instances, use the user interface to join each client to a game session. How this is done depends on how the functionality was added to the client.

If you are using the example game client, it has the following characteristics:

- A login component for the player. When connecting to a game server in an Anywhere fleet, there is no player validation. You can enter any value to enter the game session.
- A simple user interface to enter the game. When a client attempts to enter a game, it automatically searches for an active game session with an available player slot. If no game session is available, the client requests a new game session. If a game session is available, the client requests to enter the available game session. When testing the game with multiple clients simultaneously, the first client starts the game session, and the remaining clients automatically join the existing game session.
- Game sessions with four slots for players. You can start up to four instances of the client simultaneously, and they will all join the same game session.

Starting from a server executable (optional)

You can compile and start the game server executable for testing in an Anywhere fleet.

1. Compile a server executable using the Unity standard compilation process. In **File, Build Settings**, change the platform to **Dedicated Server** and compile.
2. Obtain a short-term authentication token by running the AWS CLI [get-compute-auth-token](#) command with your fleet ID and AWS region. The fleet ID is displayed in **Connect to a fleet** in Anywhere when you create the fleet. The AWS region is displayed in **Define your profile** when you select your active profile.

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

3. Inicie o executável recém-criado do servidor de jogos a partir de uma linha de comando e transmita um token de autenticação válido.

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

Plugin para Unity: implante seu jogo em uma EC2 frota gerenciada

Nesse fluxo de trabalho, você usa o plug-in para preparar seu jogo para hospedagem em recursos computacionais baseados em nuvem que são gerenciados pelo Amazon GameLift Servers. Você adiciona o código do jogo do cliente e do servidor para Amazon GameLift Servers fins de funcionalidade e, em seguida, carrega a versão do servidor no Amazon GameLift Servers serviço para hospedagem. Quando esse fluxo de trabalho estiver concluído, você terá servidores de jogo em execução na nuvem e um cliente de jogo funcional que pode se conectar a eles.

Para iniciar o EC2 fluxo de trabalho Amazon GameLift Servers gerenciado da Amazon:

- No menu principal do editor Unity, escolha Amazon GameLift Server e selecione Host with Managed EC2. Este fluxo de trabalho apresenta um processo de seis etapas para integrar, compilar, implantar e iniciar os componentes do jogo.

Definir seu perfil

Escolha o perfil que você deseja usar ao seguir esse fluxo de trabalho. O perfil selecionado afeta todas as etapas do fluxo de trabalho. Todos os recursos que você cria estão associados à AWS conta do perfil e são colocados na AWS região padrão do perfil. As permissões do usuário do perfil determinam seu acesso aos AWS recursos e ações.

1. Selecione um perfil na lista suspensa de perfis disponíveis. Se você ainda não tem um perfil ou deseja criar um novo, acesse o Amazon GameLift Servers menu e escolha Definir perfis de AWS conta.
2. Se o status do bootstrap não for “Ativo”, escolha o perfil do Bootstrap e espere que o status mude para “Ativo”.

Integre seu jogo com Amazon GameLift Servers

Nessa tarefa, você atualizará o código do cliente e do servidor em seu projeto de jogo.

- Os servidores de jogos devem ser capazes de se comunicar com o Amazon GameLift Servers serviço para receber solicitações para iniciar uma sessão de jogo, fornecer informações de conexão da sessão de jogo e relatar o status.
- Os clientes do jogo devem obter informações sobre sessões de jogo, entrar ou iniciar sessões e obter informações de conexão para entrar em um jogo.

Note

Se você importou o jogo de exemplo, poderá ignorar esta etapa. Os ativos do jogo de exemplo já têm o código necessário para o servidor e o cliente.

Integrar o código do servidor

Ao usar seu próprio projeto de jogo com cenas personalizadas, use o código de exemplo fornecido para adicionar o código de servidor necessário ao seu projeto de jogo. Se integrou o projeto de jogo para testes com uma frota Anywhere, você já concluiu as instruções nesta etapa.

1. Nos arquivos do seu projeto de jogo, abra a pasta `Assets/Scripts/Server`. Se não existir, crie-a.
2. Vá para o GitHub repositório [aws/ amazon-gamelift-plugin-unity](https://github.com/aws/amazon-gamelift-plugin-unity) e abra o caminho. `Samples~/SampleGame/Assets/Scripts/Server`
3. Localize o arquivo `GameLiftServer.cs` e copie-o para a pasta `Server` do projeto de jogo. Ao compilar um executável de servidor, use esse arquivo como destino da compilação.

O código de exemplo inclui esses elementos mínimos necessários, que usam o SDK do servidor Amazon GameLift Servers C# (versão 5):

- Inicializa um cliente de Amazon GameLift Servers API. A chamada `initSdk ()` com parâmetros do servidor é necessária para uma frota Anywhere. Amazon GameLift Servers Essas configurações são definidas automaticamente para uso no plug-in.

- Implementa as funções de retorno de chamada necessárias para responder às solicitações do Amazon GameLift Servers serviço, incluindo `OnStartGameSessionOnProcessTerminate`, e, `onHealthCheck`
- Chamadas `ProcessReady( )` com uma porta designada para notificar o Amazon GameLift Servers serviço quando o processo do servidor estiver pronto para hospedar sessões de jogo.

Se quiser personalizar o código do servidor de exemplo, consulte estes recursos:

- [Adicione Amazon GameLift Servers ao seu servidor de jogos](#)
- [SDK 5.x do servidor C# para Amazon GameLift Servers -- Ações](#)

Integrar o código do cliente

Para clientes de jogos que se conectam a servidores de jogos baseados em nuvem, é uma prática recomendada usar um serviço de back-end do lado do cliente para fazer chamadas para o Amazon GameLift Servers serviço, em vez de fazer as chamadas diretamente do cliente do jogo.

No fluxo de trabalho do plug-in para hospedagem em uma EC2 frota gerenciada, cada cenário de implantação inclui um serviço de back-end pré-criado que inclui os seguintes componentes:

- Um conjunto de funções do Lambda e tabelas do DynamoDB usadas para solicitar sessões de jogo e recuperar informações sobre sessões de jogo. Esses componentes usam um gateway da API como proxy.
- Um grupo de usuários do Amazon Cognito que gera jogadores exclusivos IDs e autentica conexões de jogadores.

Para usar esses componentes, o cliente de jogo precisa da funcionalidade de enviar solicitações ao serviço backend para fazer o seguinte:

- Crie um usuário jogador no grupo de usuários do AWS Cognito e autentique-o.
- Entrar em uma sessão de jogo e receber informações de conexão.
- Entrar em um jogo usando matchmaking.

Use os recursos a seguir como guia.

- Integre o cliente com a [GameLiftCoreApi](#) classe no GitHub repositório [amazon-gamelift-plugin-unityaws/](#). Essa classe fornece controles para autenticação do jogador e para recuperar informações de sessões de jogo.
- Para ver os exemplos de integrações de jogos, acesse o GitHub repositório [amazon-gamelift-plugin-unityaws/](#),. Samples~/SampleGame/Assets/Scripts/Client/GameLiftClient.cs
- [Adicione Amazon GameLift Servers ao seu cliente de jogo.](#)

Selecionar o cenário de implantação

Nesta etapa, você escolhe a solução de hospedagem de jogos que deseja implantar no momento. Você poderá ter várias implantações do seu jogo, usando qualquer um dos cenários.

- Frota de região única: implanta seu servidor de jogo em uma única frota de recursos de hospedagem na região padrão AWS do perfil ativo. Esse cenário é um bom ponto de partida para testar a integração do servidor AWS e a configuração de compilação do servidor. Ele implanta os seguintes recursos:
 - frota (sob demanda) da AWS com a versão do seu servidor de jogos instalada e em execução.
 - Grupo de usuários e cliente do Amazon Cognito para permitir que os jogadores se autentiquem e iniciem um jogo.
 - Autorizador de gateway de API que vincula o grupo de usuários a APIs
 - Web ACL para limitar chamadas excessivas de jogadores para o gateway da API.
 - Gateway de API + função do Lambda para os jogadores solicitarem um slot de jogo. Essa função é chamada `CreateGameSession()` se nenhuma estiver disponível.
 - Gateway de API + função do Lambda para que os jogadores obtenham informações de conexão para sua solicitação de jogo.
- FlexMatch frota: implanta seu servidor de jogo em um conjunto de frotas e configura um FlexMatch matchmaker com regras para criar partidas de jogadores. Este cenário usa hospedagem spot de baixo custo com uma estrutura de várias frotas e vários locais para uma disponibilidade duradoura. Esta abordagem é útil quando se está pronto para começar a projetar um componente de matchmaker para a solução de hospedagem. Nesse cenário, você criará os recursos básicos para essa solução, que poderão ser personalizados posteriormente, conforme necessário. Ele implanta os seguintes recursos:
 - FlexMatch configuração de matchmaking e regras de matchmaking definidas para aceitar solicitações de jogadores e formar partidas.

- Três frotas da AWS com sua versão de servidor de jogo instalada e funcionando em vários locais. Inclui duas frotas spot e uma frota sob demanda como backup.
- Fila de posicionamento de sessões de jogo da AWS que atende às solicitações de partidas propostas, encontrando o melhor recurso de hospedagem possível (com base na viabilidade, custo, latência do jogador etc.) e iniciando uma sessão de jogo.
- Grupo de usuários e cliente do Amazon Cognito para permitir que os jogadores se autentiquem e iniciem um jogo.
- Autorizador de gateway de API que vincula o grupo de usuários a APIs
- Web ACL para limitar chamadas excessivas de jogadores para o gateway da API.
- Gateway de API + função do Lambda para os jogadores solicitarem um slot de jogo. Esta função chama `StartMatchmaking()`.
- Gateway de API + função do Lambda para que os jogadores obtenham informações de conexão para sua solicitação de jogo.
- Tabelas do Amazon DynamoDB para armazenar ingressos de criação de parcerias para jogadores e informações da sessão do jogo.
- Tópico do SNS + Função Lambda para `GameSessionQueue` lidar com eventos.

Definir parâmetros de jogo

Nesta etapa, você descreve seu jogo para o qual fazer AWS o upload.

- Nome do jogo: forneça um nome significativo para o projeto de jogo. Esse nome é usado dentro do plug-in.
- Nome da frota: forneça um nome significativo para sua EC2 frota gerenciada. Amazon GameLift Servers usa esse nome (junto com o ID da frota) ao listar recursos no AWS console.
- Nome da compilação: forneça um nome significativo para a compilação do seu servidor. AWS usa esse nome para se referir à cópia da compilação do seu servidor que é carregada Amazon GameLift Servers e usada para implantações.
- Parâmetros de inicialização: insira instruções opcionais para execução ao iniciar o executável do servidor em uma instância de EC2 frota gerenciada. O tamanho máximo é de 1024 caracteres.
- Pasta do servidor do jogo: forneça o caminho para uma pasta local contendo a compilação do servidor.
- Arquivo do servidor do jogo: especifique o nome do arquivo executável do servidor.

Implantar um cenário

Nesta etapa, você implanta seu jogo em uma solução de hospedagem em nuvem com base no cenário de implantação escolhido. Esse processo pode levar vários minutos enquanto a AWS valida a compilação do servidor, provisiona recursos de hospedagem, instala o servidor de jogos, inicia os processos do servidor e os prepara para hospedar sessões de jogo.

Para iniciar a implantação, escolha Implantar CloudFormation. É possível acompanhar o status de sua hospedagem de jogos aqui. Para obter informações mais detalhadas, você pode entrar no console AWS de gerenciamento AWS e visualizar notificações de eventos. Certifique-se de fazer login usando a mesma conta, usuário e AWS região do perfil de usuário ativo no plug-in.

Quando a implantação estiver concluída, você terá seu servidor de jogo instalado em uma AWS EC2 instância. Pelo menos um processo do servidor está em execução e pronto para iniciar uma sessão de jogo.

Iniciar o cliente do jogo

Quando sua frota for implantada com sucesso, você terá servidores de jogos em execução e disponíveis para hospedar sessões de jogo. Agora você pode compilar o cliente, iniciá-lo, conectar-se para participar da sessão do jogo.

1. Configure o cliente do jogo. Nesta etapa, você solicita que o plug-in atualize um ativo `GameLiftClientSettings` para o projeto de jogo. O plug-in usa esse recurso para armazenar determinadas informações que seu cliente de jogo precisa para se conectar ao Amazon GameLift Servers serviço.
 - a. Se você não importou nem inicializou o jogo de exemplo, crie um ativo `GameLiftClientSettings`. No menu principal do editor Unity, escolha Assets, Create, Amazon GameLift, Client Settings. Se você criar várias cópias `GameLiftClientSettings` em seu projeto, o plug-in detectará isso automaticamente e notificará qual ativo o plug-in atualizará.
 - b. Em Launch Game, escolha Configurar cliente: Aplicar EC2 configurações gerenciadas. Essa ação atualiza as configurações do cliente do jogo para usar a EC2 frota gerenciada que você acabou de implantar.
2. Compile o cliente do jogo. Compile um executável de cliente usando o processo de compilação padrão do Unity. Em Arquivo, Configurações de compilação, altere a plataforma para Windows, Mac, Linux. Se você importou o jogo de exemplo e inicializou as configurações, a lista de compilações e o destino da compilação serão atualizados automaticamente.

3. Inicie o executável recém-criado do cliente do jogo. Para começar a jogar, inicie de duas a quatro instâncias do cliente e use a interface de usuário em cada uma para participar de uma sessão de jogo.

Se estiver usando o cliente de jogo de exemplo, ele tem as seguintes características:

- Um componente de login do jogador. Ao se conectar a um servidor de jogo em uma frota Anywhere, não há validação do jogador. Você poderá inserir qualquer valor para entrar na sessão do jogo.
- Uma interface de usuário simples para entrar no jogo. Quando um cliente tenta entrar em um jogo, ele procura automaticamente por uma sessão de jogo ativa com uma vaga de jogador disponível. Se nenhuma sessão de jogo estiver disponível, o cliente solicitará uma nova sessão de jogo. Se uma sessão de jogo estiver disponível, o cliente solicitará entrar na sessão de jogo disponível. Quando testar o jogo com vários clientes simultâneos, o primeiro cliente inicia a sessão do jogo, e os clientes restantes ingressam automaticamente na sessão de jogo existente.
- Sessões de jogo com quatro vagas para jogadores. Você pode iniciar até quatro instâncias de clientes de jogo simultaneamente, e elas entrarão na mesma sessão de jogo.

Plugin para Unity: implante seu jogo em uma frota de contêineres gerenciada

Use esse fluxo de trabalho de plug-in guiado para criar uma imagem de contêiner para seu servidor de jogos e implantá-la em uma solução de hospedagem baseada em contêiner. Quando você concluir esse fluxo de trabalho com êxito, seu servidor de jogos em contêiner estará sendo executado na nuvem e você poderá usar o plug-in para iniciar um cliente de jogo, conectar-se a uma sessão de jogo e jogar o jogo.

Antes de começar

Esse fluxo de trabalho pressupõe que você tenha concluído as seguintes tarefas.

- Integre o código do servidor do jogo com o SDK Amazon GameLift Servers do servidor. nosso servidor de jogos hospedado deve ser capaz de se comunicar com o Amazon GameLift Servers serviço para que ele possa responder às solicitações para iniciar novas sessões de jogo e relatar o status da sessão do jogo. Se você não concluiu essa tarefa, recomendamos que você siga primeiro o fluxo de trabalho do plug-in Host with Anywhere. Para obter orientação sobre como preparar o código do servidor do jogo, consulte [Integrar o código do servidor](#). Para uma frota

de contêineres gerenciada, você deve integrar seu jogo com o SDK do servidor versão 5.2 ou superior.

 Note

Se você estiver usando o mapa do jogo de inicialização, essa tarefa já foi feita para você.

- Package o executável do seu servidor de jogos para rodar no Linux.
- Reúna arquivos para implantar com a criação do seu servidor de jogos. Em sua máquina local, crie um diretório de trabalho para organizar os arquivos, que serão incorporados à imagem do contêiner do seu servidor de jogos. Isso pode incluir dependências do jogo, um script para iniciar servidores de jogos e outros processos ao iniciar um contêiner, etc.
- Integre seu código de cliente de jogo com Amazon GameLift Servers o. Uma forma de concluir essa tarefa é adicionar um ativo de amostra (incluído no plug-in) que já esteja integrado. Para obter orientação sobre como preparar o código do cliente do jogo, consulte [Integrar o código do cliente](#).
- Instale o Docker em sua máquina local. Você precisa instalar essa ferramenta se quiser que o plug-in crie imagens de contêiner para você e as envie para um repositório ECR. Como alternativa, você pode realizar essas tarefas manualmente e instruir o plug-in a usar uma imagem de contêiner existente. Para obter mais informações sobre como criar sua imagem manualmente, consulte [Criar uma imagem de contêiner para Amazon GameLift Servers](#).

Para iniciar o fluxo de trabalho de contêineres Amazon GameLift Servers gerenciados:

- Na barra de ferramentas principal do editor Unity, escolha o Amazon GameLift Servers menu e selecione Managed Containers. Essa ação abre a página do plug-in Host with Managed Containers, que apresenta um step-by-step processo para criar uma imagem de contêiner com a versão do seu servidor de jogos, implantá-la em uma frota de contêineres e iniciar seu jogo.

Etapa 0: defina seu perfil

Esta seção exibe seu perfil de usuário atualmente selecionado. Verifique se o perfil de usuário atual é o que você deseja usar para esse fluxo de trabalho. Todos os recursos que você cria nesse fluxo de trabalho são associados à AWS conta do perfil e colocados na AWS região padrão do perfil. As permissões do usuário do perfil determinam seu acesso aos AWS recursos e ações.

Talvez seja necessário modificar o perfil de usuário selecionado se:

- Nenhum perfil está selecionado no momento.
- Você deseja selecionar um perfil diferente ou criar um novo perfil.
- Você precisa inicializar o perfil selecionado (se o status do bootstrap estiver inativo).

Para definir ou alterar o perfil de usuário selecionado

- No Amazon GameLift Servers menu, escolha Credenciais de AWS acesso aberto.

Etapa 1: avaliar a prontidão do contêiner

Antes de implantar seu servidor de jogo em uma frota de contêineres, você deve empacotá-lo em uma imagem de contêiner e armazená-lo em um repositório Amazon ECR. O plug-in pode concluir essas tarefas para você ou você pode fazer essas tarefas manualmente. Nesta etapa, forneça informações sobre o status da imagem do contêiner e do repositório ECR.

Use as perguntas de avaliação para dizer ao plug-in quais etapas ele precisa seguir:

- Crie uma nova imagem de contêiner. Se você escolher essa opção, a próxima etapa solicitará a localização do diretório de compilação do servidor do jogo e do executável de compilação. O plug-in usa um modelo Dockerfile (fornecido por Amazon GameLift Servers) e o configura automaticamente para o seu jogo. Você pode ver o modelo em [Crie uma imagem de contêiner para Amazon GameLift Servers](#). Depois de escolher essa opção, indique onde você deseja que o plugin armazene a nova imagem:
 - Crie um novo repositório Amazon ECR e envie a imagem do contêiner para ele. O plug-in cria um repositório ECR privado usando a AWS conta e o padrão Região da AWS no perfil de usuário selecionado.
 - Envie a imagem do contêiner para um repositório Amazon ECR criado anteriormente. Se você escolher essa opção, a próxima etapa solicitará que você selecione um repositório Amazon ECR existente em uma lista. A lista inclui todos os repositórios Amazon ECR da AWS conta e do padrão Região da AWS no perfil de usuário selecionado. Você pode selecionar um repositório público ou privado.
- Use uma imagem de contêiner existente. Se você criou uma imagem manualmente, recomendamos que você use o modelo Dockerfile fornecido pela Amazon GameLift Servers, que está disponível em [Crie uma imagem de contêiner para Amazon GameLift Servers](#). Depois de escolher essa opção, indique onde a imagem está localizada.

- Uma imagem gerada pelo Docker armazenada localmente. Se você escolher essa opção, o plug-in cria um novo repositório privado do Amazon ECR e envia o arquivo de imagem local para ele. A próxima etapa solicitará um ID de imagem, que o plug-in usa para localizar o arquivo de imagem.
- Uma imagem de contêiner que já está armazenada em um repositório Amazon ECR. Se você escolher essa opção, a próxima etapa solicitará que você selecione um repositório e uma imagem existentes do Amazon ECR em uma lista. A lista inclui todos os repositórios Amazon ECR da AWS conta e do padrão Região da AWS no perfil de usuário selecionado. Você pode selecionar um repositório público ou privado.

Etapa 2: Configurar a implantação da imagem

Nesta etapa, forneça as informações de que o plug-in precisa para implantar sua imagem de contêiner em uma frota de contêineres. Essa etapa solicita as seguintes informações:

- A localização da compilação do seu servidor de jogos, da imagem do contêiner ou do repositório Amazon ECR, com base nas suas seleções na Etapa 1.
- O cenário a ser usado para a implantação de seus contêineres gerenciados.
- Configurações de implantação opcionais. Esta seção tem configurações que o plug-in usa por padrão. Você pode modificá-los ou manter os valores padrão
 - O nome do jogo é definido como o nome do seu projeto de jogo por padrão. Todos os AWS recursos criados pelo plug-in fazem referência ao valor do nome do jogo.
 - O intervalo de portas, o limite de memória e o limite de vCPU são definições de configuração para a frota de contêineres. Para obter mais informações sobre a personalização desses valores, consulte o intervalo [Configurar conexões de rede](#) de portas de conexão e os limites [Defina limites de recursos](#) de recursos.
- A tag de imagem de contêiner é usada para categorizar suas imagens de contêiner no Amazon ECR. O valor padrão é `unity-gamelift-plugin`.

Opções de cenário de implantação

Frota de contêineres de região única

Esse cenário implanta seu servidor de jogo em uma única frota de contêineres. É um bom ponto de partida para testar a integração do servidor AWS e a configuração do contêiner. Ele implanta os seguintes recursos.

- Amazon GameLift ServersA definição de grupo de contêineres descreve como implantar e executar suas imagens de contêineres em uma frota de contêineres.
- Amazon GameLift Serversfrota de contêineres (sob demanda) com seu contêiner de servidor de jogos instalado e em execução, com alias.
- Grupo de usuários e cliente do Amazon Cognito para permitir que os jogadores se autentiquem e iniciem um jogo.
- Autorizador do API Gateway que vincula o grupo de usuários a APIs
- Lista de controle de acesso à Web (ACL) para limitar chamadas excessivas de jogadores para o API Gateway.
- Serviço de back-end para fazer solicitações ao Amazon GameLift Servers serviço em nome de clientes de jogos, como solicitar sessões de jogos e participar de jogos:
 - API Gateway + função Lambda para os jogadores solicitarem um slot para uma sessão de jogo. Essa função é chamada `CreateGameSession()` se nenhum slot aberto estiver disponível.
 - API Gateway + função Lambda para que os jogadores obtenham informações de conexão para sua solicitação de jogo.

Frota de contêineres de uma única região com FlexMatch

Esse cenário implanta seu servidor de jogo em uma frota de contêineres, configura o posicionamento da sessão de jogo e FlexMatch configura o matchmaking. Esse cenário é útil quando você está pronto para começar a criar um matchmaker personalizado para sua solução de hospedagem. Use esse cenário para criar os recursos básicos para essa solução, que você pode personalizar posteriormente, conforme necessário. Ele implanta os seguintes recursos:

- Amazon GameLift Serversdefinição de grupo de contêineres que descreve como implantar e executar suas imagens de contêineres em uma frota de contêineres.
- Amazon GameLift Serversfrota de contêineres (sob demanda) com seu contêiner de servidor de jogos instalado e em execução, com alias.
- FlexMatchconfiguração de matchmaking e regras de matchmaking definidas para aceitar solicitações de jogadores e formar partidas.
- Amazon GameLift Serversfila de sessões de jogo que atende às solicitações de partidas propostas, encontrando o melhor recurso de hospedagem possível (com base na viabilidade, custo, latência do jogador etc.) e iniciando uma sessão de jogo.
- Grupo de usuários e cliente do Amazon Cognito para permitir que os jogadores se autentiquem e iniciem um jogo.

- Autorizador do API Gateway que vincula o grupo de usuários a APIs
- Lista de controle de acesso à Web (ACL) para limitar chamadas excessivas de jogadores para o API Gateway.
- Serviço de back-end para fazer solicitações ao Amazon GameLift Servers serviço em nome de clientes de jogos, como solicitar sessões de jogos e participar de jogos:
 - API Gateway + função Lambda para os jogadores solicitarem um slot para uma sessão de jogo. Essa função é chamada `StartMatchmaking()` se nenhum slot aberto estiver disponível.
 - API Gateway + função Lambda para que os jogadores obtenham informações de conexão para sua solicitação de jogo.
- Tabelas do DynamoDB para armazenar tíquetes de matchmaking para jogadores e informações da sessão do jogo.
- Tópico do Amazon SNS + função Lambda para lidar com eventos. `GameSessionQueue`

Implantar frota de contêineres

Quando a configuração da frota estiver concluída, escolha o botão **Implantar frota de contêineres** para iniciar a implantação. Esse processo pode levar vários minutos enquanto o plug-in cria uma imagem de contêiner e a envia para o ECR, provisiona recursos de hospedagem para a frota de contêineres, implanta a frota e outros AWS recursos para o cenário de solução de hospedagem selecionado.

Ao iniciar uma implantação, você pode acompanhar o progresso de cada etapa. Dependendo da sua configuração, as etapas podem incluir o seguinte:

- Configurando a imagem do contêiner
- Criação de um repositório do Amazon ECR
- Criando uma imagem e enviando para o Amazon ECR
- Criação de definição de grupo de contêiner
- Criando frota de contêineres

Para obter informações de implantação mais detalhadas, escolha **Exibir** no console AWS de gerenciamento. Quando a frota de contêineres atinge o status ativo, a frota está executando ativamente contêineres com processos de servidor prontos para hospedar sessões de jogo.

Quando a implantação estiver concluída, você terá uma frota de contêineres em funcionamento pronta para hospedar sessões de jogo e aceitar conexões de jogadores.

Você não pode interromper uma implantação em andamento. Se a implantação entrar em um estado ruim ou falhar, você poderá recomeçar usando a opção Redefinir implantação.

Cliente de lançamento

Neste ponto, você concluiu todas as tarefas para iniciar e jogar seu jogo multijogador hospedado no Amazon GameLift Servers. Para jogar seu jogo, escolha Start Client para iniciar uma instância local do seu cliente de jogo.

- Se você implantou o cenário de frota única, abra uma instância do seu cliente de jogo com um jogador e entre no mapa do servidor para se movimentar. Você pode abrir uma segunda instância do cliente do jogo para adicionar um segundo jogador ao mesmo mapa de jogo do servidor.
- Se você implantou o FlexMatch cenário, a solução de hospedagem espera que pelo menos dois clientes do jogo façam solicitações de matchmaking. Abra pelo menos duas instâncias do seu cliente de jogo com um jogador. Os dois jogadores serão emparelhados e solicitados a participar de uma sessão de jogo para a partida.

Atualize uma frota de contêineres

Se você implantou com sucesso uma solução de hospedagem de contêineres gerenciados, poderá usar o recurso Atualizar implantação. Essa opção permite que você atualize as configurações de uma frota de contêineres implantada, sem precisar criar uma nova frota.

Ao atualizar uma implantação, você pode implantar uma imagem de contêiner com uma versão de servidor de jogo diferente, alterar o repositório do Amazon ECR, escolher um cenário de implantação diferente e personalizar as configurações opcionais.

Quando você estiver pronto para implantar suas alterações, escolha Atualizar. O tempo necessário para uma atualização de implantação é semelhante ao de uma implantação completa. Para obter informações detalhadas de implantação, escolha Exibir no console AWS de gerenciamento.

Limpe os recursos implantados

Como prática recomendada, limpe os AWS recursos de sua solução de contêineres gerenciados assim que você não precisar mais deles. Você pode continuar incorrendo em custos com esses recursos se não os remover.

Exclua os seguintes recursos:

- Pilha gerenciada de recursos de contêineres. Os recursos nessa pilha dependem do cenário de implantação selecionado. Para excluir a pilha inteira, use o AWS CloudFormation console. As pilhas geradas a partir do Amazon GameLift Servers plug-in usam a seguinte convenção de nomenclatura: `GameLiftPluginForUnity-{GameName}-Containers`. Aguarde a conclusão do processo de exclusão da pilha antes de iniciar uma nova implantação de contêineres gerenciados no plug-in. Para obter mais informações, consulte [Excluir uma pilha do CloudFormation console](#).
- Repositório Amazon ECR. Se você usou o plug-in para criar um repositório para sua imagem de contêiner, talvez queira excluir os repositórios que não são mais necessários. Você não precisa excluir um repositório antes de redefinir uma implantação de contêineres gerenciados. Se você atualizar ou redefinir uma implantação, o plug-in usará automaticamente o mesmo repositório, a menos que seja orientado a usar outro. Para obter mais informações, consulte [Excluir um repositório privado no Amazon ECR](#).

Amazon GameLift Serversplugin para Unity (servidor SDK 4.x)

Note

Este tópico fornece informações para uma versão anterior do Amazon GameLift Servers plug-in para Unity. A versão 1.0.0 (lançada em 2021) usa o SDK do servidor para Amazon GameLift Servers 4.x ou anterior. Para obter a documentação sobre a versão mais recente do plug-in, que usa o SDK 5.x do servidor e é compatível com Amazon GameLift Servers Anywhere, consulte [Amazon GameLift Serversplugin para Unity \(servidor SDK 5.x\)](#).

Amazon GameLift Servers fornece ferramentas para preparar seus servidores de jogos multijogador para serem executados no Amazon GameLift Servers. O Amazon GameLift Servers plug-in para Unity facilita a integração do Amazon GameLift Servers aos seus projetos de jogos Unity e a implantação de recursos do Amazon GameLift Servers para hospedagem na nuvem. Use o plug-in do Unity para acessar as APIs do Amazon GameLift Servers e implantar modelos de AWS CloudFormation para cenários de jogos comuns.

Depois de configurar o plug-in, você pode experimentar a [amostra do Amazon GameLift Servers Unity](#) em GitHub.

Tópicos

- [Integre-se ao Amazon GameLift Servers com um projeto de servidor de jogos Unity](#)

- [Integre-se Amazon GameLift Servers com um projeto de cliente de jogo Unity](#)
- [Instalar e configurar o plug-in](#)
- [Testar o jogo localmente](#)
- [Implantar um cenário](#)
- [Integre jogos com o Amazon GameLift Servers Unity](#)
- [Importar e executar um jogo de amostra](#)

Integre-se Amazon GameLift Servers com um projeto de servidor de jogos Unity

Note

Este tópico fornece informações para uma versão anterior do Amazon GameLift Servers plug-in para Unity. A versão 1.0.0 (lançada em 2021) usa o SDK do servidor para Amazon GameLift Servers 4.x ou anterior. Para obter a documentação sobre a versão mais recente do plug-in, que usa o SDK 5.x do servidor e é compatível com Amazon GameLift Servers Anywhere, consulte [Amazon GameLift Servers plug-in para Unity \(servidor SDK 5.x\)](#).

Este tópico ajuda você a preparar seu servidor de jogos personalizado para hospedagem Amazon GameLift Servers. O servidor do jogo deve ser capaz de notificar Amazon GameLift Servers sobre seu status, iniciar e interromper sessões de jogo quando solicitado e realizar outras tarefas. Para obter mais informações, consulte [Adicione Amazon GameLift Servers ao seu servidor de jogos](#).

Pré-requisitos

Antes de integrar o servidor de jogos, execute as tarefas a seguir:

- [Configure uma função de serviço do IAM para Amazon GameLift Servers](#)
- [Instalar e configurar o plug-in](#)

Configure um novo processo de servidor

Note

Este tópico se refere ao Amazon GameLift Servers plug-in para Unity versão 1.0.0, que usa o SDK do servidor 4.x ou anterior.

Configure a comunicação Amazon GameLift Servers e informe que o processo do servidor está pronto para hospedar uma sessão de jogo.

1. Inicialize o SDK do servidor ao chamar `InitSDK()`.
2. Para preparar o servidor para aceitar uma sessão de jogo, chame `ProcessReady()` com a porta de conexão e os detalhes da localização da sessão de jogo. Inclua os nomes das funções de retorno de chamada que o Amazon GameLift Servers serviço invoca, como, `OnGameSession()`, `OnGameSessionUpdate()`, `OnProcessTerminate()`, `OnHealthCheck()`. Amazon GameLift Servers pode levar alguns minutos para fornecer um retorno de chamada.
3. Amazon GameLift Servers atualiza o status do processo do servidor para `ACTIVE`.
4. Amazon GameLift Servers liga `onHealthCheck` periodicamente.

O exemplo de código a seguir mostra como configurar um processo de servidor simples com 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;
}
```

Iniciar uma sessão de jogo

Note

Este tópico se refere ao Amazon GameLift Servers plug-in para Unity versão 1.0.0, que usa o SDK do servidor 4.x ou anterior.

Depois que a inicialização do jogo estiver concluída, você poderá iniciar uma sessão de jogo.

1. Implemente a função `onStartGameSession` de retorno de chamada. Amazon GameLift Servers invoca esse método para iniciar uma nova sessão de jogo no processo do servidor e receber conexões de jogadores.
2. Para ativar uma sessão de jogo, chame `ActivateGameSession()`. Para obter mais informações sobre o SDK, consulte [SDK do servidor C# para Amazon GameLift Servers 4.x -- Ações](#).

O exemplo de código a seguir ilustra como iniciar uma sessão de jogo com 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();  
}
```

Encerrar uma sessão de jogo

Note

Este tópico se refere ao Amazon GameLift Servers plug-in para Unity versão 1.0.0, que usa o SDK do servidor 4.x ou anterior.

Notifique Amazon GameLift Servers quando uma sessão de jogo estiver terminando. Como melhor prática, desligue os processos do servidor após a conclusão das sessões de jogo para reciclar e atualizar os recursos de hospedagem.

1. Configure uma função chamada `onProcessTerminate` para receber solicitações Amazon GameLift Servers e chamadas `ProcessEnding()`.
2. O status do processo muda para `TERMINATED`.

O exemplo a seguir descreve como finalizar um processo para uma sessão de jogo.

```
var processEndingOutcome = GameLiftServerAPI.ProcessEnding();  
  
if (processReadyOutcome.Success)  
    Environment.Exit(0);  
  
// otherwise, exit with error code  
Environment.Exit(errorCode);
```

Crie a compilação do servidor e faça o upload para Amazon GameLift Servers

Note

Este tópico se refere ao Amazon GameLift Servers plug-in para Unity versão 1.0.0, que usa o SDK do servidor 4.x ou anterior.

Depois de integrar seu servidor de jogos Amazon GameLift Servers, carregue os arquivos de compilação em uma frota para que ela Amazon GameLift Servers possa implantá-los na hospedagem de jogos. Para obter mais informações sobre como fazer o upload do seu servidor para Amazon GameLift Servers, consulte [Implemente uma criação de servidor personalizada para Amazon GameLift Servers hospedagem](#).

Integre-se Amazon GameLift Servers com um projeto de cliente de jogo Unity

Note

Este tópico fornece informações para uma versão anterior do Amazon GameLift Servers plug-in para Unity. A versão 1.0.0 (lançada em 2021) usa o SDK do servidor para Amazon GameLift Servers 4.x ou anterior. Para obter a documentação sobre a versão mais recente do plug-in, que usa o SDK 5.x do servidor e é compatível com Amazon GameLift Servers Anywhere, consulte [Amazon GameLift Servers plugin para Unity \(servidor SDK 5.x\)](#)

Este tópico ajuda você a configurar um cliente de jogo para se conectar a sessões de jogos Amazon GameLift Servers hospedadas por meio de um serviço de back-end. Use Amazon GameLift Servers APIs para iniciar matchmaking, solicitar colocação em sessão de jogo e muito mais.

Adicione código ao projeto de serviço de back-end para permitir a comunicação com o Amazon GameLift Servers serviço. Um serviço de back-end gerencia toda a comunicação do cliente do jogo com o GameLift serviço. Para obter mais informações sobre esses serviços de back-end, consulte .

Um servidor de back-end executa as seguintes tarefas do cliente do jogo:

- Personalize a autenticação para seus jogadores.
- Solicitar informações sobre sessões de jogos ativas do serviço do Amazon GameLift Servers.
- Cria uma sessão de jogo.

- Adicionar um jogador a uma sessão de jogo existente.
- Remover um jogador de uma sessão de jogo existente.

Tópicos

- [Pré-requisitos](#)
- [Inicializar um cliente de jogo](#)
- [Crie a sessão do jogo em uma frota específica](#)
- [Adicionar jogadores às sessões de jogo](#)
- [Remova um jogador de uma sessão de jogo](#)

Pré-requisitos

Antes de configurar a comunicação do servidor do jogo com o Amazon GameLift Servers cliente, conclua as seguintes tarefas:

- [Configurar um Conta da AWS](#)
- [Instalar e configurar o plug-in](#)
- [Integre-se Amazon GameLift Servers com um projeto de servidor de jogos Unity](#)
- [Configurando uma frota de hospedagem com Amazon GameLift Servers](#)

Inicializar um cliente de jogo

Note

Este tópico se refere ao Amazon GameLift Servers plug-in para Unity versão 1.0.0, que usa o SDK do servidor 4.x ou anterior.

Adicione código para inicializar um cliente de jogo. Execute esse código no lançamento, é necessário para outras Amazon GameLift Servers funções.

1. Inicializar `AmazonGameLiftClient`. Chame `AmazonGameLiftClient` com uma configuração de cliente padrão ou uma configuração personalizada. Para obter mais informações sobre como configurar um cliente, consulte [Configurar Amazon GameLift Servers em um serviço de back-end](#).

2. Gere um ID de jogador exclusivo para cada jogador se conectar a uma sessão de jogo. Para obter mais informações, consulte [Gerar jogador IDs](#).

Os exemplos a seguir mostram como configurar um Amazon GameLift Servers cliente.

```
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);
    }
}
```


Crie a sessão do jogo em uma frota específica

Note

Este tópico se refere ao Amazon GameLift Servers plug-in para Unity versão 1.0.0, que usa o SDK do servidor 4.x ou anterior.

Adicione um código para iniciar novas sessões de jogos nas frotas implantadas e disponibilizá-las para jogadores. Depois Amazon GameLift Servers de criar a nova sessão de jogo e retornar uma `GameSession`, você pode adicionar jogadores a ela.

- Faça uma solicitação para uma nova sessão de jogo.
 - Se seu jogo usa frotas, chame `CreateGameSession()` com um ID de frota ou alias, um nome de sessão e o número máximo de jogadores simultâneos no jogo.
 - Se o seu jogo usa filas, chame `StartGameSessionPlacement()`.

O exemplo a seguir mostra como criar uma sessão de jogo.

```
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;
}
```

Adicionar jogadores às sessões de jogo

Note

Este tópico se refere ao Amazon GameLift Servers plug-in para Unity versão 1.0.0, que usa o SDK do servidor 4.x ou anterior.

Depois Amazon GameLift Servers de criar a nova sessão de jogo e devolver um `GameSession` objeto, você pode adicionar jogadores a ela.

1. Reserve um slot de jogador em uma sessão de jogo criando uma nova sessão de jogador. Use `CreatePlayerSession` ou `CreatePlayerSessions` com o ID da sessão do jogo e um ID exclusivo para cada jogador.
2. Conecte-se à sessão do jogo. Recupere o objeto `PlayerSession` para obter as informações de conexão da sessão do jogo. É possível usar essas informações para estabelecer uma conexão direta com o processo do servidor:
 - a. Use a porta especificada e o nome DNS ou endereço IP do processo do servidor.
 - b. Use o nome DNS e a porta de suas frotas. O nome DNS e a porta serão necessários se suas frotas tiverem a geração de certificados TLS habilitada.
 - c. Faça referência ao ID da sessão do jogador. O ID da sessão do jogador é necessário se o servidor do jogo validar as conexões de entrada do jogador.

Os exemplos a seguir demonstram como reservar uma vaga de jogador em uma sessão de jogo.

```
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;
```

```
}
```

O código a seguir ilustra como conectar um jogador à sessão do jogo.

```
public bool ConnectPlayer(int playerId, string playerIdSessionId)
{
    //Call ConnectPlayer with player ID and player session ID.
    return server.ConnectPlayer(playerId, playerIdSessionId);
}
```

Remova um jogador de uma sessão de jogo

 Note

Este tópico se refere ao Amazon GameLift Servers plug-in para Unity versão 1.0.0, que usa o SDK do servidor 4.x ou anterior.

Você poderá remover os jogadores da sessão de jogo quando eles saírem do jogo.

1. Notifique o Amazon GameLift Servers serviço de que um jogador se desconectou do processo do servidor. Chame `RemovePlayerSession` com o ID da sessão do jogador.
2. Verifique se isso `RemovePlayerSession` retorna `Success`. Em seguida, Amazon GameLift Servers altera o slot do jogador para ficar disponível, o que Amazon GameLift Servers pode ser atribuído a um novo jogador.

O exemplo a seguir ilustra como remover uma sessão de jogador.

```
public void DisconnectPlayer(int playerId)
{
    //Receive the player session ID.
    string playerIdSessionId = playerIdSessions[playerId];
    var outcome = GameLiftServerAPI.RemovePlayerSession(playerIdSessionId);
    if (outcome.Success)
    {
        Debug.Log (":) PLAYER SESSION REMOVED");
    }
    else
    {

```

```
        Debug.Log(":(PLAYER SESSION REMOVE FAILED. RemovePlayerSession()  
        returned " + outcome.Error.ToString());  
    }  
}
```

Instalar e configurar o plug-in

Note

Este tópico se refere ao Amazon GameLift Servers plug-in para Unity versão 1.0.0, que usa o SDK do servidor 4.x ou anterior.

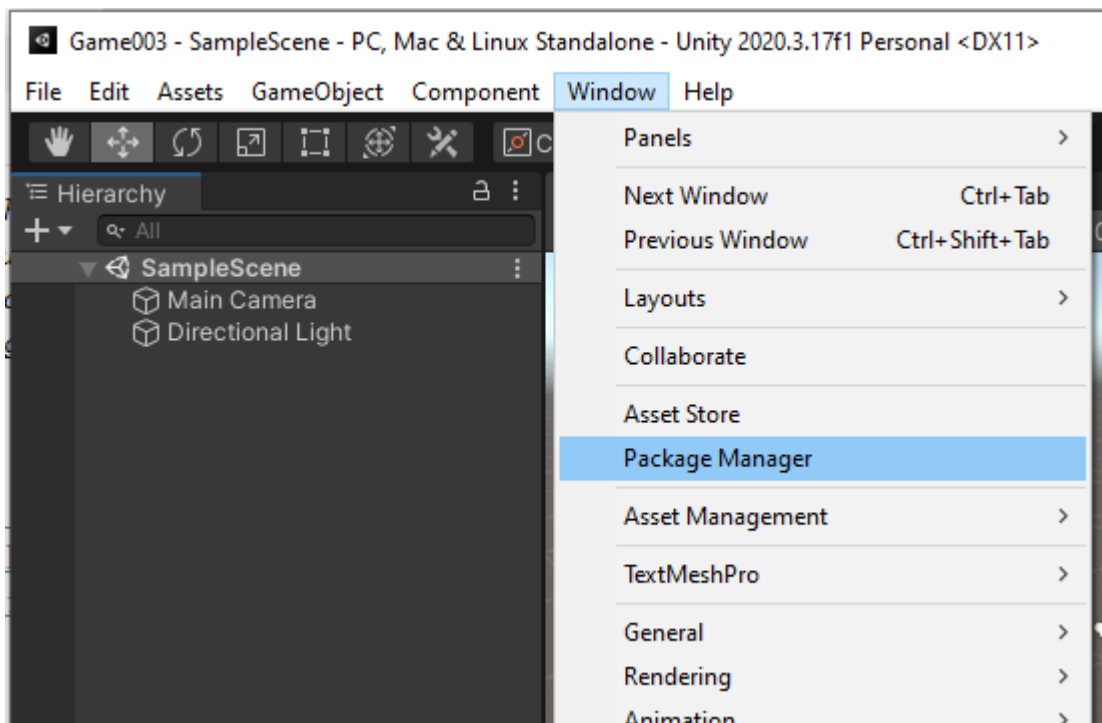
Esta seção descreve como baixar, instalar e configurar o Amazon GameLift Servers plug-in para Unity, versão 1.0.0.

Pré-requisitos

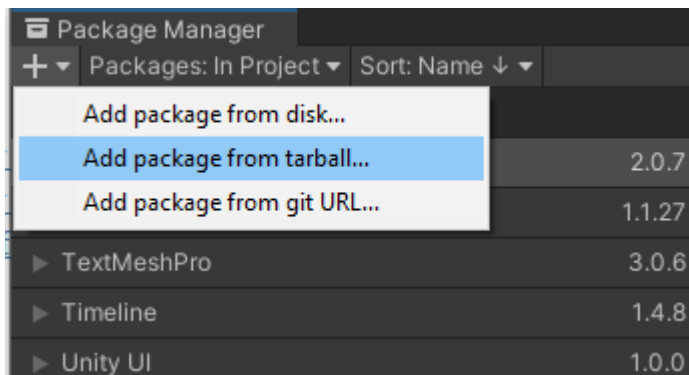
- Unity para Windows 2019.4 LTS, Windows 2020.3 LTS ou Unity para macOS
- Versão atual do Java
- Versão atual do .NET 4.x

Para fazer download e instalar o plug-in para Unity

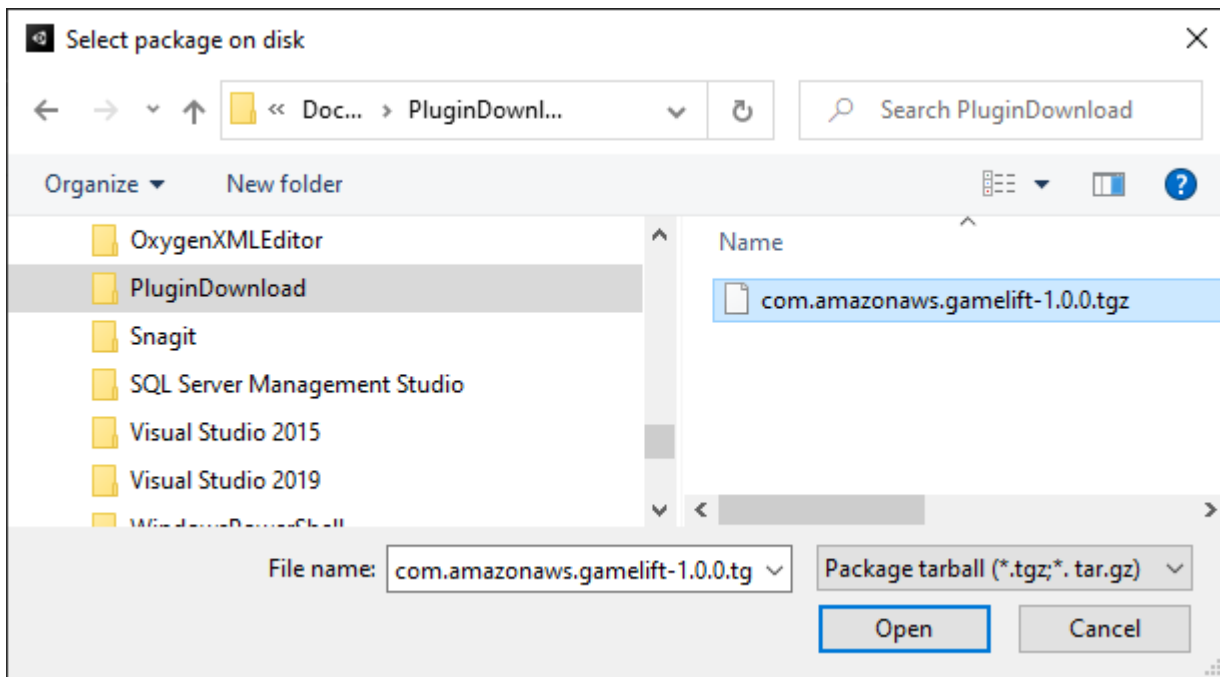
1. Baixe o Amazon GameLift Servers plug-in para Unity. Você pode encontrar a versão mais recente na página do [Amazon GameLift Servers plug-in para o repositório Unity](#). Na [versão mais recente](#), escolha Ativos e, em seguida, baixe o arquivo `com.amazonaws.gamelift-version.tgz`.
2. Inicie o Unity e escolha um projeto.
3. Na barra de navegação superior, em Window, escolha Gerenciador de pacotes:



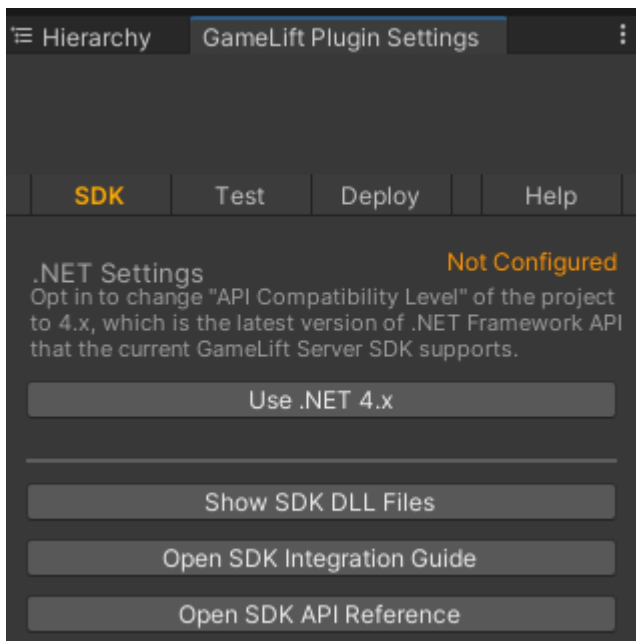
4. No guia Gerenciador de pacotes, escolha + e, em seguida, escolha Adicionar pacote do tarball...:



5. Na janela Selecionar pacotes no disco, navegue até a pasta com `.amazonaws.gamelift`, escolha o arquivo com `.amazonaws.gamelift-version.tgz` e escolha Abrir:



6. Depois que o Unity carrega o plug-in, Amazon GameLift Servers aparece como um novo item no menu Unity. Pode demorar alguns minutos para instalar e recompilar scripts. A guia Configurações do Amazon GameLift Servers plug-in é aberta automaticamente.



7. No painel SDK, escolha Usar o .NET 4.x.

Quando configurado, o status muda de Não configurado para Configurado.

Testar o jogo localmente

Note

Este tópico se refere ao Amazon GameLift Servers plug-in para Unity versão 1.0.0, que usa o SDK do servidor 4.x ou anterior.

Use Amazon GameLift Servers Local para executar Amazon GameLift Servers em seu dispositivo local. Você pode usar o Amazon GameLift Servers Local para verificar as alterações do código em segundos, sem uma conexão de rede.

Configurar testes locais

1. Na janela do plug-in para Unity, escolha o guia Teste.
2. No painel Teste, escolha Baixar Amazon GameLift Servers local. O plug-in para Unity abre uma janela do navegador e baixa o arquivo `GameLift_06_03_2021.zip` para sua pasta de downloads.

O download inclui o SDK do servidor C#, arquivos de origem do .NET e um componente do .NET compatível com o Unity.

3. Não descompacte o arquivo `GameLift_06_03_2021.zip` obtido por download.
4. Na janela Configurações do Amazon GameLift Servers plug-in, escolha Amazon GameLift ServersLocal Path, navegue até a pasta descompactada, escolha o arquivo **`GameLiftLocal.jar`** e escolha Abrir.

Quando configurado, o status do teste local muda de Não configurado para Configurado.

5. Verifique o status do JRE. Se o status for Não configurado, escolha Baixar o JRE e instale a versão recomendada do Java.

Depois de instalar e configurar o ambiente Java, o status muda para Configurado.

Executar o jogo local

1. No guia Plug-in para Unity, escolha o guia Teste.
2. No painel Teste, escolha Abrir interface de teste local.

3. Na janela Teste local, especifique um Caminho executável do servidor. Selecione... para selecionar o caminho e o nome do executável do seu aplicativo de servidor.
4. Na janela Teste local, especifique uma porta local GL.
5. Escolha Implantar e executar para implantar e executar o servidor.
6. Para parar seu servidor de jogos, escolha Parar ou feche as janelas do servidor de jogos.

Implantar um cenário

Note

Este tópico se refere ao Amazon GameLift Servers plug-in para Unity versão 1.0.0, que usa o SDK do servidor 4.x ou anterior.

Um cenário usa um AWS CloudFormation modelo para criar os recursos necessários para implantar uma solução de hospedagem em nuvem para seu jogo. Esta seção descreve os cenários Amazon GameLift Servers fornecidos e como usá-los.

Pré-requisitos

Para implantar o cenário, você precisa de uma função do IAM para o Amazon GameLift Servers serviço. Para obter informações sobre como criar uma função para Amazon GameLift Servers, consulte [Configurar um Conta da AWS](#).

Cada cenário exige permissões para os seguintes recursos:

- Amazon GameLift Servers
- Amazon S3
- AWS CloudFormation
- API Gateway
- AWS Lambda
- AWS WAFV2
- Amazon Cognito

Cenários

Note

Este tópico se refere ao Amazon GameLift Servers plug-in para Unity versão 1.0.0, que usa o SDK do servidor 4.x ou anterior.

O Amazon GameLift Servers plug-in para Unity inclui os seguintes cenários:

Somente Autenticação

Esse cenário cria um serviço de back-end de jogos que realiza a autenticação do jogador sem a capacidade do servidor de jogos. O modelo cria os seguintes recursos na sua conta:

- Um grupo de usuários do Amazon Cognito para armazenar informações de autenticação de jogadores.
- Um AWS Lambda manipulador baseado em endpoint REST do Amazon API Gateway que inicia jogos e visualiza as informações de conexão do jogo.

Frota de uma única região

Esse cenário cria um serviço de back-end de jogos com uma única Amazon GameLift Servers frota. Ele cria os recursos a seguir:

- Um grupo de usuários do Amazon Cognito para que um jogador se autentique e inicie um jogo.
- Um AWS Lambda manipulador para pesquisar uma sessão de jogo existente com um espaço de jogador aberto na frota. Se não conseguir encontrar um slot aberto, ele cria uma nova sessão de jogo.

Frota multirregional com fila e marcador de jogos personalizado

Esse cenário forma partidas usando Amazon GameLift Servers filas e um matchmaker personalizado para agrupar os jogadores mais velhos na piscina de espera. Ele cria os recursos a seguir:

- Um tópico do Amazon Simple Notification Service que Amazon GameLift Servers publica mensagens para. Para obter mais informações sobre tópicos e notificações do SNS, consulte [Configurar notificações de eventos para o posicionamento de sessões de jogo](#).

- Uma função do Lambda que é invocada pela mensagem que comunica os detalhes do posicionamento e da conexão do jogo.
- Uma tabela do Amazon DynamoDB para armazenar detalhes de posicionamento e conexão do jogo. As chamadas `GetGameConnection` são lidas dessa tabela e retornam as informações de conexão para o cliente do jogo.

Identifique frotas com uma fila e um marcador de jogos personalizado

Esse cenário forma correspondências usando Amazon GameLift Servers filas e um matchmaker personalizado e configura três frotas. Ele cria os recursos a seguir:

- Duas frotas Spot que contêm diferentes tipos de instâncias para fornecer durabilidade em caso de indisponibilidade do Spot.
- Uma frota sob demanda que atua como backup para as outras frotas Spot. Para obter mais informações sobre como projetar suas frotas, consulte [Personalize seu Amazon GameLift Servers EC2 frotas gerenciadas](#).
- Uma Amazon GameLift Servers fila para manter a disponibilidade do servidor alta e o custo baixo. Para obter mais informações e melhores práticas sobre filas, consulte [Personalize uma fila de sessões de jogo](#).

FlexMatch

Este cenário usa FlexMatch, um serviço gerenciado de matchmaking, para combinar jogadores. Para obter mais informações sobre FlexMatch, consulte [O que é Amazon GameLift Servers FlexMatch](#).

Este cenário cria os recursos a seguir:

- Uma função do Lambda para criar um ticket de marcador de jogos depois de receber solicitações `StartGame`.
- Uma função Lambda separada para ouvir os eventos da FlexMatch partida.

Para evitar cobranças desnecessárias Conta da AWS, remova os recursos criados por cada cenário depois de terminar de usá-los. Exclua a AWS CloudFormation pilha correspondente.

Atualizar as credenciais da AWS

Note

Este tópico se refere ao Amazon GameLift Servers plug-in para Unity versão 1.0.0, que usa o SDK do servidor 4.x ou anterior.

O Amazon GameLift Servers plug-in para Unity requer credenciais de segurança para implantar um cenário. Você poderá criar novas credenciais ou usar credenciais existentes.

Para obter mais informações sobre como configurar credenciais, consulte [Entendendo e obtendo suas AWS credenciais](#).

Para atualizar as AWS credenciais

1. No Unity, no guia Plug-in para Unity, escolha o guia Implantar.
2. No painel Implantar, escolha Credenciais da AWS .
3. Você pode criar novas AWS credenciais ou escolher credenciais existentes.
 - Para criar credenciais, escolha Criar novo perfil de credenciais e, em seguida, especifique o Nome do novo perfil, ID da chave de acesso da AWS , chave secreta da AWS e Região da AWS.
 - Para escolher as credenciais existentes, escolha Escolher perfil de credenciais existente e, em seguida, selecione um nome de perfil e Região da AWS.
4. Na janela Atualizar AWS Credenciais, escolha Atualizar Perfil de Credenciais.

Atualizar a inicialização da conta

Note

Este tópico se refere ao Amazon GameLift Servers plug-in para Unity versão 1.0.0, que usa o SDK do servidor 4.x ou anterior.

A localização do bootstrap é um bucket do Amazon S3 usado durante a implantação. Ele é usado para armazenar ativos do servidor de jogos e outras dependências. A Região da AWS que você escolher para o bucket deve ser a mesma região que você usará para a implantação do cenário.

Para obter mais informações sobre os buckets do Amazon S3, consulte [Criar, configurar e trabalhar com os buckets do Amazon Simple Storage Service](#).

Para atualizar a localização do bootstrap da conta

1. No Unity, no guia Plug-in para Unity, escolha a guia Implantar.
2. No painel Implantar, escolha Atualizar conta Bootstrap.
3. Na janela Account Bootstrapping, você escolhe um bucket do Amazon S3 existente ou cria um novo bucket do Amazon S3:
 - Para escolher um bucket existente, escolha Escolher bucket do Amazon S3 existente e Atualizar para salvar sua seleção.
 - Escolha Criar o bucket do Amazon S3 para criar um bucket do Amazon Simple Storage Service e, em seguida, escolha uma política. A política especifica quando o bucket do Amazon S3 expirará. Escolha Criar para criar o bucket.

Implantar um cenário de jogo

 Note

Este tópico se refere ao Amazon GameLift Servers plug-in para Unity versão 1.0.0, que usa o SDK do servidor 4.x ou anterior.

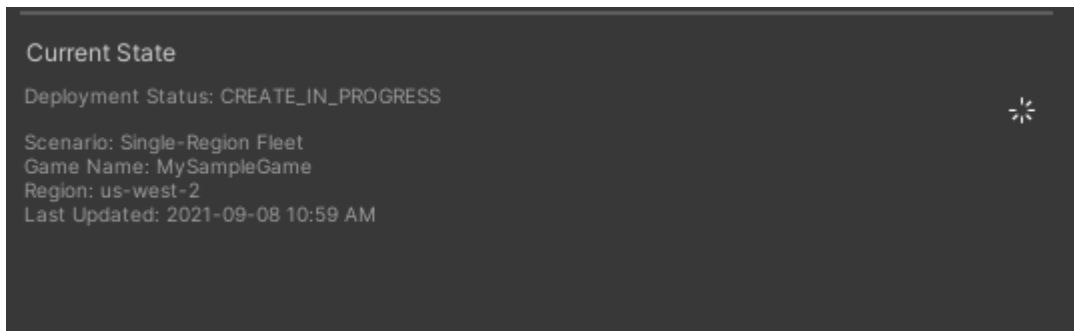
Você pode usar um cenário para testar seu jogo Amazon GameLift Servers. Cada cenário usa um modelo AWS CloudFormation para criar uma pilha com os recursos necessários. A maioria dos cenários exige um servidor de jogo executável e um caminho de construção. Ao implantar o cenário, Amazon GameLift Servers copia os ativos do jogo para o local de bootstrap como parte da implantação.

Você deve configurar AWS as credenciais e um bootstrap de AWS conta para implantar um cenário.

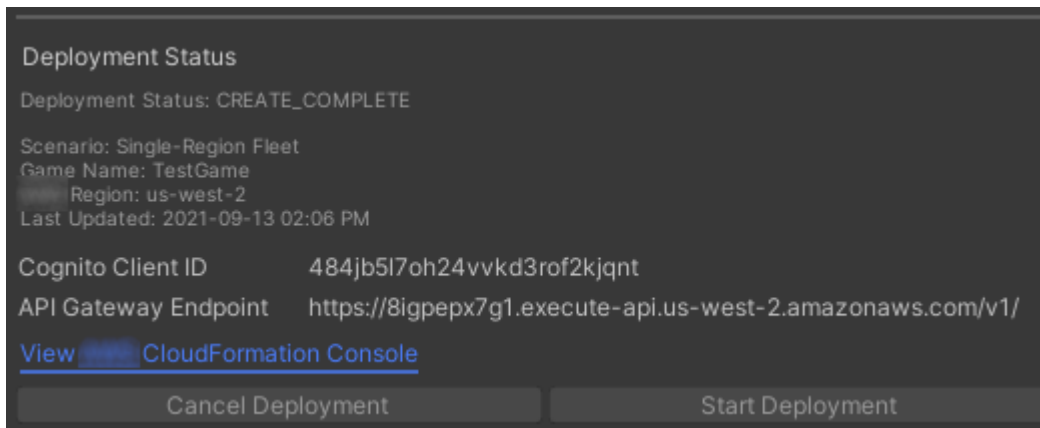
Para implantar um cenário

1. No Unity, no guia Plug-in para Unity, escolha o guia Implantar.
2. No painel Implantar, escolha Abrir interface de implantação.
3. Na janela Implantar, escolha um cenário.

4. Insira um Nome de Jogo. Essa opção deve ser exclusiva. O nome do jogo faz parte do nome da AWS CloudFormation pilha quando você implanta o cenário.
5. Escolha o Caminho da pasta de compilação do servidor de jogo. O caminho da pasta de compilação aponta para a pasta que contém o executável e as dependências do servidor.
6. Escolha o Caminho do arquivo .exe de compilação do servidor de jogo. O caminho do arquivo executável de compilação aponta para o executável do servidor do jogo.
7. Escolha Iniciar implantação para começar a implantar um cenário. Você poderá acompanhar o status da atualização na janela Implantação em Estado atual. Os cenários podem levar vários minutos para serem implantados.



8. Quando a implantação do cenário é concluída, o Estado atual é atualizado para incluir o ID do cliente Cognito e a Endpoint do API Gateway, que você pode copiar e colar no jogo.



9. Para atualizar as configurações do jogo, no menu Unity, escolha Ir para configurações de conexão do cliente. Isso exibe um guia Inspector no lado direito da tela do Unity.
10. Desmarque o Modo de teste local.
11. Insira o Endpoint do API Gateway e o ID do cliente Cognito. Escolha o mesmo Região da AWS que você usou para a implantação do cenário. Em seguida, você poderá reconstruir e executar o cliente do jogo usando os recursos do cenário implantado.

Excluindo recursos criados pelo cenário

Note

Este tópico se refere ao Amazon GameLift Servers plug-in para Unity versão 1.0.0, que usa o SDK do servidor 4.x ou anterior.

Para excluir os recursos criados para o cenário, exclua a pilha AWS CloudFormation correspondente.

Para excluir recursos criados pelo cenário

1. Na janela do Amazon GameLift Servers plug-in para Unity Deployment, selecione View AWS CloudFormation Console para abrir o AWS CloudFormation console.
2. No AWS CloudFormation console, escolha Pilhas e, em seguida, escolha a pilha que inclui o nome do jogo especificado durante a implantação.
3. Escolha Excluir para excluir a pilha. Pode levar alguns minutos para excluir uma pilha. Depois de AWS CloudFormation excluir a pilha usada pelo cenário, seu status muda para. ROLLBACK_COMPLETE

Integre jogos com o Amazon GameLift Servers Unity

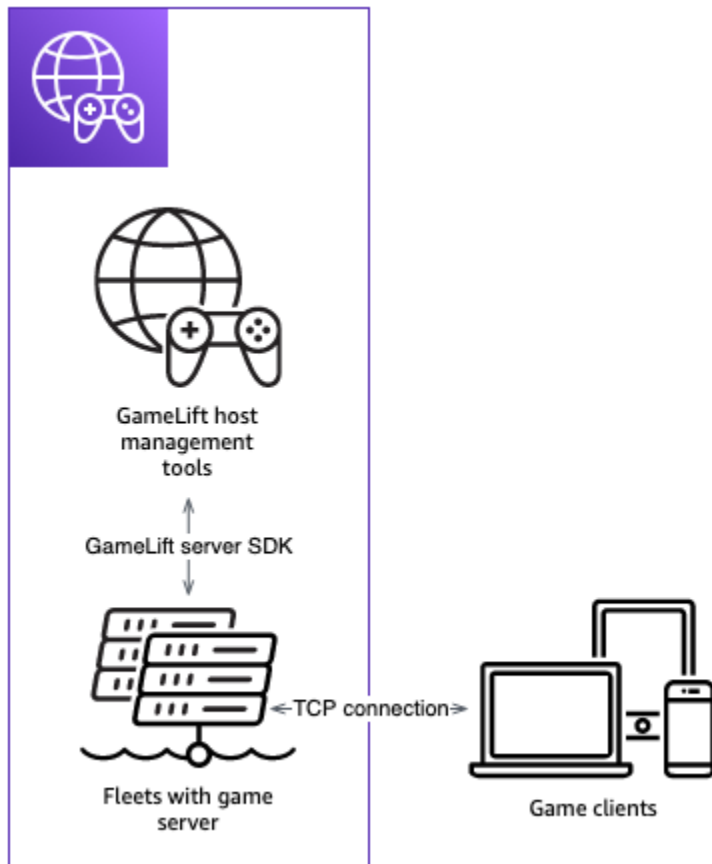
Note

Este tópico se refere ao Amazon GameLift Servers plug-in para Unity versão 1.0.0, que usa o SDK do servidor 4.x ou anterior.

Integre seu jogo Unity Amazon GameLift Servers ao concluir as seguintes tarefas:

- [Integre-se Amazon GameLift Servers com um projeto de servidor de jogos Unity](#)
- [Integre-se Amazon GameLift Servers com um projeto de cliente de jogo Unity](#)

O diagrama a seguir mostra um exemplo de fluxo de integração de um jogo. No diagrama, uma frota com o servidor do jogo é implantada em. Amazon GameLift Servers O cliente do jogo se comunica com o servidor do jogo, que se comunica com. Amazon GameLift Servers



Importar e executar um jogo de amostra

Note

Este tópico se refere ao Amazon GameLift Servers plug-in para Unity versão 1.0.0, que usa o SDK do servidor 4.x ou anterior.

O Amazon GameLift Servers plug-in para Unity inclui um jogo de amostra que você pode usar para explorar os fundamentos da integração do seu jogo. Amazon GameLift Servers Nesta seção, você cria o cliente e o servidor do jogo e depois testa localmente usando o Amazon GameLift Servers Local.

Pré-requisitos

- [Configurar um Conta da AWS](#)
- [Instalar e configurar o plug-in](#)

Criar e executar a aplicação de exemplo

Note

Este tópico se refere ao Amazon GameLift Servers plug-in para Unity versão 1.0.0, que usa o SDK do servidor 4.x ou anterior.

Configure os arquivos do servidor do jogo de amostra.

1. No Unity, no menu, escolha e Amazon GameLift Servers, em seguida, escolha Importar amostra de jogo.
2. Na janela Importar amostra de jogo, escolha Importar para importar o jogo, seus ativos e dependências.
3. Crie o servidor do jogo. No Unity, no menu, escolha e, em seguida Amazon GameLift Servers, escolha Aplicar configurações de compilação do Windows Sample Server ou Aplicar configurações de compilação do macOS Sample Server. Depois de definir as configurações do servidor do jogo, o Unity recompila os ativos.
4. No Unity, no menu, escolha Arquivo e, em seguida, escolha Compilar. Escolha Compilação de servidor, escolha Compilar e, em seguida, escolha uma pasta de compilação específica para arquivos do servidor.

O Unity cria o servidor de jogo de amostra, colocando os ativos executáveis e necessários na pasta de compilação especificada.

Criar e executar a aplicação do jogo de exemplo

Note

Este tópico se refere ao Amazon GameLift Servers plug-in para Unity versão 1.0.0, que usa o SDK do servidor 4.x ou anterior.

Configure os arquivos do cliente do jogo de amostra.

1. No Unity, no menu, escolha e, em seguida Amazon GameLift Servers, escolha Aplicar configurações de compilação do Windows Sample Client ou Aplicar configurações de

compilação do cliente de amostra do macOS. Depois que as configurações do cliente do jogo forem definidas, o Unity recompilará os ativos.

2. No Unity, no menu, selecione Ir para configurações do cliente. Isso exibirá um guia Inspector no lado direito da tela do Unity. Na guia Configurações do Amazon GameLift Servers cliente, selecione Modo de teste local.
3. Crie o cliente do jogo. No Unity, no menu, escolha Arquivo. Confirme se a opção Compilação de servidor não está marcada, escolha Compilação e selecione uma pasta de compilação específica para arquivos do cliente.

O Unity cria o cliente de jogo de amostra, colocando os ativos executáveis e necessários na pasta de criação do cliente especificada.

4. Você não construiu o servidor e o cliente do jogo. Nas próximas etapas, você executa o jogo e vê como ele interage com Amazon GameLift Servers.

Teste o jogo de amostra localmente

Note

Este tópico se refere ao Amazon GameLift Servers plug-in para Unity versão 1.0.0, que usa o SDK do servidor 4.x ou anterior.

Execute o jogo de amostra que você importou usando o Amazon GameLift Servers Local.

1. Inicie o servidor do jogo. No Unity, no guia Plug-in para Unity, escolha o guia Implantar.
2. No painel Teste, escolha Abrir interface de teste local.
3. Na janela Teste local, especifique um Caminho de arquivo .exe do servidor de jogos. O caminho deve incluir o nome do executável. Por exemplo, `.C:/MyGame/GameServer/MyGameServer.exe`
4. Escolha Implantar e executar. O plug-in para Unity inicia o servidor do jogo e abre uma janela de registro Amazon GameLift Servers local. A janela contém mensagens de log, incluindo mensagens enviadas entre o servidor do jogo e o Amazon GameLift Servers Local.
5. Inicie o cliente do jogo. Encontre o local de compilação com o cliente do jogo de amostra e escolha o arquivo executável.
6. No Jogo de Amazon GameLift Servers Amostra, forneça um e-mail e uma senha e, em seguida, escolha Entrar. O e-mail e a senha não são validados nem usados.

7. No Jogo Amazon GameLift Servers de Amostra, escolha Iniciar. O cliente do jogo procura uma sessão de jogo. Se ele não conseguir encontrar uma sessão, ele cria uma. O cliente do jogo então inicia a sessão do jogo. Você poderá ver a atividade do jogo nos registros.

Exemplos de registros do servidor de jogos

```
...
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
```

Exemplos de registros Amazon GameLift Servers locais

```
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: gameSessionId: "arn:aws:gamelift:local::gamesession/
fleet-123/gsess-59f6cc44-4361-42f5-95b5-fdb5825c0f3d"
```

Desligar um processo do servidor

Note

Este tópico se refere ao Amazon GameLift Servers plug-in para Unity versão 1.0.0, que usa o SDK do servidor 4.x ou anterior.

Depois de terminar seu jogo de amostra, desligue o servidor no Unity.

1. No cliente do jogo, escolha Sair ou feche a janela para parar o cliente do jogo.
2. No Unity, na janela Teste local, escolha Parar ou feche as janelas do servidor de jogos para parar o servidor.

Roteiro de desenvolvimento para hospedagem com gerenciamento Amazon GameLift Servers EC2

Este roteiro orienta você sobre como desenvolver uma solução de EC2 hospedagem Amazon GameLift Servers gerenciada para seu jogo multijogador. Amazon GameLift Servers oferece várias opções de hospedagem de jogos; para obter mais informações sobre essas opções, consulte [Soluções do Amazon GameLift Servers](#).

Com a hospedagem Amazon GameLift Servers gerenciada, seu servidor de jogos é hospedado em recursos de computação virtual Nuvem AWS baseados que Amazon GameLift Servers possuem e operam com base em sua configuração. Você obtém a segurança, a confiabilidade e a disponibilidade global das instâncias do Amazon Elastic Compute Cloud (Amazon EC2) que são ainda mais otimizadas para uso com hospedagem de jogos multijogador. Amazon GameLift Servers simplifica o gerenciamento de hospedagem com ferramentas como implantações automáticas de servidores, gerenciamento do ciclo de vida e auto-scaling de capacidade.

Uma solução Amazon GameLift Servers gerenciada é composta pelos seguintes componentes:

- Uma ou mais frotas Amazon GameLift Servers gerenciadas, que usam instâncias do Amazon Elastic Compute Cloud EC2 (Amazon) otimizadas para hospedagem de jogos multijogador.
- Um servidor de jogos construído, integrado ao SDK do servidor para ser Amazon GameLift Servers implantado em todas as frotas.
- Um cliente de jogo e serviço de back-end, integrado ao AWS SDK, para interagir com o Amazon GameLift Servers serviço e solicitar sessões de jogo.
- Uma Amazon GameLift Servers fila para colocar novas sessões de jogo com servidores de jogos disponíveis em todas as frotas.
- (Opcional) Um FlexMatch matchmaker para criar partidas multijogador e configurar sessões de jogo para elas.

Este roteiro apresenta um caminho simplificado para colocar seu jogo multijogador em funcionamento com sucesso com a hospedagem Amazon GameLift Servers gerenciada EC2 . Depois de configurar os componentes necessários, você pode continuar a iterar no desenvolvimento de jogos e personalizar a solução de hospedagem. À medida que você se aproxima do lançamento, consulte [Preparando seu jogo para o lançamento com Amazon GameLift Servers hospedagem](#) para obter ajuda na preparação da solução de hospedagem para uso em nível de produção.

i Comece com o Amazon GameLift Servers plug-in para Unreal Engine e Unity

Para uma implantação mais rápida, experimente o [Amazon GameLift Servers plug-in](#) para Unreal Engine e Unity. Ele fornece fluxos de trabalho de interface de usuário guiados para implantar rapidamente seu servidor de jogos com configuração mínima, para que você possa testar os componentes do jogo em ação. Depois, você pode aproveitar essa base para criar uma solução de hospedagem personalizada para o seu jogo. Consulte mais detalhes em [Explore com o Amazon GameLift Servers plug-in](#).

Etapa 1: Prepare seu servidor de jogo para trabalhar com Amazon GameLift Servers

Adicione funcionalidade ao seu servidor de jogos para que ele possa se comunicar com o Amazon GameLift Servers serviço quando ele for implantado para hospedagem.

- Obtenha o SDK do servidor para Amazon GameLift Servers (versão 5.x) para seu projeto de jogo. O SDK do servidor está disponível em C++, C# e Go. [Baixe o SDK de um Amazon GameLift Servers servidor](#).
- Modifique o código do servidor de jogos para adicionar a funcionalidade do SDK do servidor. Para obter orientações, consulte [Integre jogos com servidores de jogos personalizados](#). No mínimo, faça o seguinte:
 - Adicione código para inicializar o Amazon GameLift Servers SDK e estabelecer uma WebSocket conexão com o Amazon GameLift Servers serviço. Use a ação `InitSdk()` do SDK do servidor.
 - Adicione código para reportar ao Amazon GameLift Servers serviço quando o processo do servidor estiver pronto para hospedar sessões de jogo. Use a ação `ProcessReady()` do SDK do servidor.
 - Implemente as funções `OnProcessTerminate()` de retorno de chamada necessárias e `OnStartGameSession()`. Com essas funções, os processos do servidor de jogos podem manter uma conexão com o Amazon GameLift Servers serviço, iniciar uma sessão de jogo quando solicitado e responder a uma solicitação para encerrar o processo do servidor de jogos.
- Adicione código para reportar ao Amazon GameLift Servers serviço quando o processo do servidor estiver encerrando uma sessão de jogo. Use a ação `ProcessEnding()` do SDK do servidor.
- Empacote a versão do servidor de jogos. Crie um script de instalação com seus arquivos de compilação, dependências e software de suporte. Consulte [Criar um pacote dos arquivos de](#)

[compilação do jogo](#). Recomendamos usar um bucket do Amazon Simple Storage Service (Amazon S3) para armazenar as versões da compilação do jogo.

- Teste a integração do servidor de jogos. Para essa tarefa, recomendamos configurar uma frota Amazon GameLift Servers Anywhere para uma estação de trabalho local, conforme descrito em [Configure testes locais com o Amazon GameLift Servers Anywhere](#). Para esta etapa, instale manualmente a compilação do servidor de jogos no dispositivo de teste e inicie um processo no servidor. Use a AWS CLI para solicitar uma nova sessão de jogo e verifique se o Amazon GameLift Servers serviço solicita com êxito que o processo do servidor inicie uma sessão de jogo.

Etapa 2: prepare o cliente do jogo para entrar em sessões de jogo hospedadas

Crie uma forma do cliente de jogo solicitar a participação em uma sessão de jogo, obter informações de conexão e conectar-se diretamente a uma sessão de jogo hospedada. A abordagem mais comum é configurar a funcionalidade do serviço de back-end que serve como intermediária entre o cliente do jogo e o serviço. Amazon GameLift Servers Essa abordagem protege os recursos de hospedagem e oferece maior controle sobre como os jogadores são colocados nas sessões de jogo.

- Crie uma funcionalidade de serviço de backend para hospedagem. O serviço de back-end se comunica com o Amazon GameLift Servers serviço e fornece informações de conexão a um cliente de jogo. Essa funcionalidade inclui iniciar sessões de jogo, colocar jogadores em jogos e recuperar informações da sessão de jogo. Para obter orientações, consulte [Integre jogos com servidores de jogos personalizados](#). No mínimo, faça o seguinte:
 - Obtenha o AWS SDK Amazon GameLift Servers e adicione-o ao seu projeto de serviço de back-end. Veja os [recursos Amazon GameLift Servers do SDK para serviços ao cliente](#).
 - Adicione código para inicializar um Amazon GameLift Servers cliente e armazenar as configurações da chave. Consulte [Configurar Amazon GameLift Servers em um serviço de back-end](#).
 - Adicione funcionalidade para chamar a ação do AWS SDK `CreateGameSession()` e fornecer informações de conexão da sessão de jogo a um cliente de jogo. Consulte [Criar uma sessão de jogo em uma frota específica](#).

Ligar `CreateGameSession()` é um ponto de partida conveniente para solicitar novas sessões de jogo. Depois de instalar um sistema de posicionamento de sessões de jogo (consulte a Etapa 3), você substituirá esse código por uma chamada para `StartGameSessionPlacement()` (ou `StartMatchmaking()` se estiver usando `FlexMatch`).

Para obter orientação sobre como projetar o serviço de backend, consulte [Projete seu serviço de cliente de jogos](#).

- Adicione a funcionalidade ao cliente de jogo que permite que os jogadores participem de uma sessão de jogo hospedada. O cliente do jogo faz solicitações para seu serviço de back-end, não diretamente para o Amazon GameLift Servers. Depois que o serviço de backend fornece as informações de conexão da sessão do jogo, o cliente de jogo se conecta diretamente à sessão do jogo para jogar.
- Teste a integração do cliente de jogo. Você pode usar a mesma frota Amazon GameLift Servers Anywhere com uma estação de trabalho local para testes.

Para um desenvolvimento iterativo rápido ou ao trabalhar com equipes de várias pessoas, recomendamos que você configure um [ambiente de teste baseado na nuvem](#). Essa solução Amazon GameLift Servers de kit de ferramentas imita o comportamento de uma frota Amazon GameLift Servers gerenciada, mas permite que você atualize as compilações do servidor de jogos com o mínimo de tempo de resposta.

Etapa 3: configure o posicionamento da sessão de jogo

Personalize como você deseja Amazon GameLift Servers processar as solicitações de uma nova sessão de jogo e localize os servidores de jogos disponíveis para hospedá-las. Amazon GameLift Servers rastreia automaticamente a disponibilidade de todos os servidores de jogos em todas as frotas. Quando um cliente de jogo envia uma solicitação para participar de uma sessão de jogo, Amazon GameLift Servers procura o “melhor posicionamento possível” com base em um conjunto de prioridades definidas, como latência, custo e disponibilidade mínimos.

- Crie uma fila de sessões de jogo para alocar uma nova sessão de jogo em servidores de jogos disponíveis. As filas são o principal mecanismo para posicionamento de sessões de jogo. Para obter orientações, consulte [Criar uma fila de sessões de jogo](#).
- No mínimo, adicione suas frotas Anywhere como destinos na fila. Todas as outras configurações são personalizações opcionais.
- No código de serviço de backend, converta a chamada **CreateGameSession()** em **StartGameSessionPlacement()**. Consulte [Criar uma sessão de jogo em uma fila com vários locais](#).
- Crie um mecanismo para notificar um cliente de jogo quando uma sessão de jogo estiver pronta para ser iniciada. Durante o desenvolvimento, você pode pesquisar o status da sessão do jogo

usando uma chamada para `DescribeGameSessionPlacement`. No entanto, antes de usar uma fila para processar grandes volumes, você precisará ativar as notificações de eventos. Consulte [Configurar notificações de eventos para o posicionamento de sessões de jogo](#).

- (Opcional) Adicione componentes de FlexMatch matchmaking. Para obter orientação, consulte o [guia do Amazon GameLift ServersFlexMatch desenvolvedor](#).

Etapa 4: crie frotas gerenciadas baseadas na nuvem

Até agora, você trabalhou com frotas autogerenciadas do Anywhere para testar e iterar os componentes do seu jogo e ajustou o posicionamento da sessão de jogo. A parte final da solução é configurar o tipo de recursos de hospedagem necessários para um sistema de produção. Para começar a planejar e configurar a produção, você deseja fazer a transição para trabalhar com uma frota Amazon GameLift Servers gerenciada.

- Package a compilação do seu servidor de jogo e faça o upload para Amazon GameLift Servers. Crie um script de instalação com seus arquivos de compilação, dependências e software de suporte. Consulte [Implemente uma criação de servidor personalizada para Amazon GameLift Servers hospedagem](#). Você pode fazer o upload da sua compilação Amazon GameLift Servers usando o console ou a AWS CLI.

Antes de fazer o upload da sua construção, decida em que frota Região da AWS você quer criar. É necessário fazer o upload da compilação para a mesma região. Para saber mais sobre como escolher a localização da frota, consulte [Localização da frota](#).

- Crie uma EC2 frota gerenciada. Quando você cria uma frota, começa Amazon GameLift Servers imediatamente a implantar sua versão de servidor de jogos para hospedagem. Você pode configurar vários aspectos de uma frota gerenciada. Para obter orientações, consulte [Crie uma EC2 frota Amazon GameLift Servers gerenciada](#). No mínimo, faça o seguinte:
 - Dê um nome à frota e especifique qual compilação de jogo enviada deve ser implantada.
 - Escolha instâncias sob demanda para a frota e selecione um tipo de instância que esteja disponível na localização da frota. As frotas spot são uma opção valiosa, mas exigem design e configuração adicionais.
 - Crie uma configuração de tempo de execução para a frota. No mínimo, especifique o caminho de inicialização para o executável do servidor de jogos.
 - Especifique as configurações de porta para permitir que o tráfego de entrada acesse os servidores de jogos.

- Adicione as frotas gerenciadas à fila. Na fila de sessões de jogo, substitua as frotas do Anywhere as frotas gerenciadas.
- Teste a hospedagem de jogos com as frotas gerenciadas. Nesse ponto, você poderá testar todo o ciclo de hospedagem, com um cliente de jogo solicitando uma sessão de jogo, obtendo informações de conexão e se conectando com sucesso a uma sessão de jogo.

Etapa 5: personalize as frotas gerenciadas

Ao se preparar para o lançamento do jogo, você precisará ajustar os recursos de hospedagem gerenciada. Algumas das decisões a serem consideradas incluem:

- Considere adicionar frotas spot para reduzir custos. Consulte [Tutorial: Crie uma Amazon GameLift Servers fila com instâncias spot](#).
- Se o servidor do jogo precisar comunicar outros AWS recursos, configure funções do IAM para gerenciar o acesso. Consulte [Comunique-se com outros AWS recursos de suas frotas](#).
- Determine onde você deseja posicionar geograficamente os servidores de jogo. Adicione locais remotos às suas frotas gerenciadas. Consulte [Personalize seu Amazon GameLift Servers EC2 frotas gerenciadas](#).
- Otimize o desempenho da frota selecionando um tipo e um tamanho de instância e configurando o tempo de execução para executar vários processos do servidor. Consulte [Gerencie como Amazon GameLift Servers lança servidores de jogos](#).
- Experimente as opções de posicionamento de sessões de jogo para frotas gerenciadas, incluindo a personalização das configurações de priorização. Consulte [Personalize uma fila de sessões de jogo](#).
- Configure o ajuste de escala automático da capacidade para atender à demanda esperada dos jogadores. Consulte [Eskalando a capacidade de hospedagem de jogos com Amazon GameLift Servers](#).
- Configure frotas em espera em outras Regiões da AWS e modifique as filas e o escalonamento automático para lidar com failovers, se necessário.
- Configure ferramentas de observabilidade de hospedagem, incluindo análises e registros em log. Consulte [Monitoramento Amazon GameLift Servers](#).
- Automatize a implantação usando [infraestrutura como código \(IaC\)](#). Consulte [Gerenciamento Amazon GameLift Servers recursos de hospedagem usando AWS CloudFormation](#).

Amazon GameLift Servers suporta o uso de AWS CloudFormation modelos para qualquer configuração específica da implantação. Você também pode usar o AWS Cloud Development Kit (AWS CDK) para definir seus Amazon GameLift Servers recursos. Para obter mais informações sobre o AWS CDK, consulte o [Guia do AWS Cloud Development Kit \(AWS CDK\) desenvolvedor](#).

Para gerenciar a implantação de suas AWS CloudFormation pilhas, recomendamos o uso de ferramentas e serviços de integração contínua e entrega contínua (CI/CD), como. AWS CodePipeline Essas ferramentas ajudam você a implantar automaticamente ou com aprovação sempre que você cria o binário do servidor de jogos. Com uma CI/CD ferramenta ou serviço, a implantação de recursos para uma nova versão do servidor de jogos pode ter a seguinte aparência:

- Compile e teste o binário do servidor de jogos.
- Faça o upload do binário para Amazon GameLift Servers.
- Implante novas frotas com a nova compilação.
- Adicione as novas frotas à fila de sessões de jogo e remova as frotas com a versão de compilação anterior.
- Quando as frotas com a versão anterior não estiverem mais hospedando sessões de jogo ativas, exclua as AWS CloudFormation pilhas dessas frotas.

Roteiro de desenvolvimento para hospedagem com contêineres Amazon GameLift Servers gerenciados

Este roteiro orienta você sobre como desenvolver uma solução de hospedagem Amazon GameLift Servers gerenciada para seus servidores de jogos em contêineres. Os contêineres gerenciados são apenas uma solução de hospedagem oferecida pela Amazon GameLift Servers. Para obter mais informações sobre as opções de hospedagem, consulte [Soluções do Amazon GameLift Servers](#).

Uma solução de contêiner gerenciado Amazon GameLift Servers tem os seguintes componentes:

- Uma ou mais frotas de contêineres, que usam instâncias do Amazon Elastic Compute Cloud EC2 (Amazon) otimizadas para hospedagem de jogos multijogador.
- Uma imagem de contêiner com a criação do seu servidor de jogo, carregada no repositório privado do Amazon Elastic Container Registry (Amazon ECR). A versão do servidor de jogos é integrada ao SDK do servidor Amazon GameLift Servers e criada para ser executada no Linux.

- Um serviço de back-end que interage com o Amazon GameLift Servers serviço em nome de seus clientes de jogos. O serviço de back-end usa a funcionalidade na API de serviço para Amazon GameLift Servers, que faz parte do AWS SDK.
- Uma fila de sessões de Amazon GameLift Servers jogo que processa solicitações de novas sessões de jogo, procura servidores de jogos disponíveis em todas as frotas e solicita que um servidor de jogos inicie uma sessão de jogo.
- (Opcional) Um FlexMatch matchmaker para criar partidas multijogador e configurar sessões de jogo para elas.

Esse roteiro apresenta um caminho simplificado para colocar seus servidores de jogos em contêineres em funcionamento com sucesso com contêineres gerenciados. Amazon GameLift Servers Depois de configurar os componentes necessários, você pode continuar a iterar no desenvolvimento de jogos e personalizar a solução de hospedagem. À medida que você se aproxima do lançamento, consulte [Preparando seu jogo para o lançamento com Amazon GameLift Servers hospedagem](#) para obter ajuda na preparação da solução de hospedagem para uso em nível de produção.

 Acelere a integração com essas ferramentas para contêineres gerenciados:

- O [kit inicial de contêineres](#) agiliza a integração e a configuração da frota. Ele adiciona recursos essenciais de gerenciamento de sessões de jogos ao seu servidor de jogos e usa modelos pré-configurados para criar uma frota de contêineres e um pipeline de implantação automatizado para seu servidor de jogos. Após a implantação, use o Amazon GameLift Servers console e as ferramentas de API para monitorar o desempenho da frota, gerenciar sessões de jogo e analisar métricas.
- Para desenvolvedores do Unreal Engine e Unity, use os [Amazon GameLift Servers plug-ins](#) para integrar seu servidor de jogos e criar uma frota de contêineres de dentro do ambiente de desenvolvimento do seu mecanismo de jogo. Os fluxos de trabalho guiados do plug-in ajudam você a criar uma solução rápida e simples com hospedagem baseada em nuvem usando contêineres gerenciados. Você pode aproveitar essa base para criar uma solução de hospedagem personalizada para seu jogo.

Etapa 1: Prepare seu servidor de jogo para trabalhar com Amazon GameLift Servers

Adicione funcionalidade ao seu servidor de jogos para que ele possa se comunicar com o Amazon GameLift Servers serviço quando for implantado para hospedagem.

- Obtenha o SDK do servidor Amazon GameLift Servers (versão 5.2 ou superior) para seu projeto de jogo. O SDK do servidor está disponível em C++, C# e Go. [Baixe o SDK do servidor para Amazon GameLift Servers](#). O SDK do servidor está disponível em C++, C# e Go.
- Modifique o código do servidor de jogos para adicionar a funcionalidade do SDK do servidor. Para obter orientações, consulte [Integre jogos com servidores de jogos personalizados](#). No mínimo, faça o seguinte:
 - Adicione código para inicializar o Amazon GameLift Servers SDK e estabelecer uma WebSocket conexão com o Amazon GameLift Servers serviço. Use a ação `InitSdk()` do SDK do servidor.
 - Adicione código para reportar ao Amazon GameLift Servers serviço quando o processo do servidor estiver pronto para hospedar sessões de jogo. Use a ação `ProcessReady()` do SDK do servidor.
 - Implemente as funções `OnStartGameSession()` de retorno de chamada necessárias e `OnProcessTerminate()`. Com essas funções, os processos do servidor de jogos podem manter uma conexão com o Amazon GameLift Servers serviço, iniciar uma sessão de jogo quando solicitado e responder a uma solicitação para encerrar o processo do servidor de jogos. Amazon GameLift Servers
 - Adicione código para reportar ao Amazon GameLift Servers serviço quando o processo do servidor estiver encerrando uma sessão de jogo. Use a ação `ProcessEnding()` do SDK do servidor.
- Empacote a versão do servidor de jogos. Crie seu servidor de jogos para rodar no Linux. Prepare a compilação e outros arquivos necessários para executar o servidor do jogo. Se você estiver desenvolvendo no Windows, essa etapa pode envolver a configuração de um espaço de trabalho Linux separado ou o uso de uma ferramenta como o subsistema Windows para Linux (WSL). Você precisará de um ambiente Linux para testar a compilação do seu servidor de jogos e também para criar e testar suas imagens de contêiner.
- Teste a integração do servidor de jogos. Verifique se o servidor de jogos integrado pode se conectar ao Amazon GameLift Servers serviço e responder às solicitações. Recomendamos configurar uma frota simples do Amazon GameLift Servers Anywhere com uma estação de trabalho local como host de teste, conforme descrito em [Configure testes locais com o Amazon GameLift Servers Anywhere](#). Instale sua versão de servidor de jogos no host de teste e inicie um processo no servidor. Use a AWS CLI para solicitar uma nova sessão de jogo e verifique se o

Amazon GameLift Servers serviço solicita com êxito que o processo do servidor inicie uma sessão de jogo.

Etapa 2: Prepare o cliente do jogo para entrar em sessões de jogo hospedadas.

Crie uma forma do cliente de jogo solicitar a participação em uma sessão de jogo, obter informações de conexão e conectar-se diretamente a uma sessão de jogo hospedada. A abordagem mais comum é configurar a funcionalidade do serviço de back-end que serve como intermediária entre o cliente do jogo e o serviço. Amazon GameLift Servers Essa abordagem protege os recursos de hospedagem e oferece maior controle sobre como os jogadores são colocados nas sessões de jogo.

- Crie uma funcionalidade de serviço de backend para hospedagem. O serviço de back-end se comunica com o Amazon GameLift Servers serviço e fornece informações de conexão a um cliente de jogo. Essa funcionalidade inclui iniciar sessões de jogo, colocar jogadores em jogos e recuperar informações da sessão de jogo. Para obter orientações, consulte [Integre jogos com servidores de jogos personalizados](#). No mínimo, faça o seguinte:
 - Obtenha o AWS SDK Amazon GameLift Servers e adicione-o ao seu projeto de serviço de back-end. Veja os [recursos Amazon GameLift Servers do SDK para serviços ao cliente](#).
 - Adicione código para inicializar um Amazon GameLift Servers cliente e armazenar as configurações da chave. Consulte [Configurar Amazon GameLift Servers em um serviço de back-end](#).
 - Adicione funcionalidade para chamar a ação do AWS SDK `CreateGameSession()` e fornecer informações de conexão da sessão de jogo a um cliente de jogo. Consulte [Criar uma sessão de jogo em uma frota específica](#).

Ligar `CreateGameSession()` é um ponto de partida conveniente para solicitar novas sessões de jogo. Depois de instalar um sistema de posicionamento de sessões de jogo (consulte a Etapa 3), você substituirá esse código por uma chamada para `StartGameSessionPlacement()` (ou `StartMatchmaking()` se estiver usando `FlexMatch`).

Para obter orientação sobre como projetar o serviço de backend, consulte [Projete seu serviço de cliente de jogos](#).

- Adicione a funcionalidade ao cliente de jogo que permite que os jogadores participem de uma sessão de jogo hospedada. O cliente do jogo faz solicitações para seu serviço de back-end, não diretamente para o. Amazon GameLift Servers Depois que o serviço de backend fornece as informações de conexão da sessão do jogo, o cliente de jogo se conecta diretamente à sessão do jogo para jogar.

- Teste a integração do cliente de jogo. Você pode usar sua frota Amazon GameLift Servers Anywhere existente com uma estação de trabalho local para testes. Use o novo serviço de back-end para solicitar uma nova sessão de jogo e verifique se: (1) o Amazon GameLift Servers serviço solicita com êxito que o processo do servidor inicie uma sessão de jogo e (2) um cliente de jogo pode se conectar à sessão de jogo.

Etapa 3: Configure o posicionamento da sessão de jogo

Personalize como você deseja Amazon GameLift Servers processar as solicitações de uma nova sessão de jogo e localize os servidores de jogos disponíveis para hospedá-las. Amazon GameLift Servers rastreia automaticamente a disponibilidade de todos os servidores de jogos em todas as frotas. Quando um cliente de jogo envia uma solicitação para participar de uma sessão de jogo, Amazon GameLift Servers procura o “melhor posicionamento possível” com base em um conjunto de prioridades definidas, como latência, custo e disponibilidade mínimos.

- Crie uma fila de sessões de jogo para alocar uma nova sessão de jogo em servidores de jogos disponíveis. As filas são o principal mecanismo para posicionamento de sessões de jogo. Para obter orientações, consulte [Criar uma fila de sessões de jogo](#).
- No mínimo, adicione suas frotas Anywhere como destinos na fila. Todas as outras configurações são personalizações opcionais.
- No código de serviço de backend, converta a chamada **CreateGameSession()** em **StartGameSessionPlacement()**. Consulte [Criar uma sessão de jogo em uma fila com vários locais](#).
- Crie um mecanismo para notificar um cliente de jogo quando uma sessão de jogo estiver pronta para ser iniciada. Durante o desenvolvimento, você pode pesquisar o status da sessão do jogo usando uma chamada para `DescribeGameSessionPlacement`. No entanto, antes de usar uma fila para processar grandes volumes, você precisará ativar as notificações de eventos. Consulte [Configurar notificações de eventos para o posicionamento de sessões de jogo](#).
- Adicionar FlexMatch matchmaking (opcional). Crie um conjunto de regras de matchmaking e crie uma configuração de matchmaking para trabalhar com sua fila de sessões de jogo. Para obter orientação sobre como configurar um sistema de matchmaking, consulte o [guia do Amazon GameLift Servers FlexMatch desenvolvedor](#).
- Teste o sistema de posicionamento. Você pode usar sua frota Amazon GameLift Servers Anywhere existente com uma estação de trabalho local para testes. Use o serviço de back-end para solicitar uma nova sessão de jogo e verifique se o Amazon GameLift Servers serviço solicita com êxito que o processo do servidor inicie uma sessão de jogo.

Etapa 4: criar uma imagem de contêiner do servidor de jogos

Depois de integrar com sucesso o servidor de jogos, crie uma imagem de contêiner com o executável do servidor de jogos. Armazene-o em um repositório do Amazon Elastic Container Registry (Amazon ECR) para uso com Amazon GameLift Servers. Para obter instruções detalhadas, consulte [Crie uma imagem de contêiner para Amazon GameLift Servers](#).

- Obtenha o modelo Dockerfile para um contêiner de servidor de jogos (fornecido por Amazon GameLift Servers). Modifique o arquivo dos arquivos de compilação do seu servidor de jogos.
- Crie uma imagem de contêiner de servidor de jogos. Trabalhando em um ambiente Linux, use a ferramenta Docker para criar sua imagem.
- Envie sua imagem de contêiner para o Amazon ECR. Crie um repositório público ou privado no Amazon ECR, usando o mesmo Conta da AWS e Região da AWS onde você planeja implantar sua frota de contêineres. Empurre a imagem do contêiner para ela.
- Teste suas imagens de contêiner usando sua frota Anywhere (opcional). Talvez você queira testar suas imagens de contêiner localmente antes de implantá-las em uma frota de contêineres hospedada na nuvem. Você pode usar sua frota Amazon GameLift Servers Anywhere existente com uma estação de trabalho local para testes. Instale e execute o contêiner do servidor do jogo e verifique se: (1) o Amazon GameLift Servers serviço solicita com êxito que o processo do servidor inicie uma sessão de jogo e (2) um cliente do jogo pode se conectar à sessão do jogo.

Etapa 5: criar uma frota de contêineres baseada em nuvem

Até agora, você trabalhou com uma frota autogerenciada do Anywhere para testar e iterar os componentes do seu jogo. A parte final da sua solução é configurar os recursos de hospedagem baseados em nuvem que você precisará para um sistema de produção. Para começar a planejar e configurar a produção, você deve configurar uma frota Amazon GameLift Servers gerenciada de contêineres e personalizá-la para produção.

- Crie definições de grupos de contêineres. As definições de grupos de contêineres descrevem a arquitetura de contêineres de uma frota e identificam quais imagens de contêiner devem ser implantadas. Consulte [Crie uma definição de grupo de contêineres para uma frota de Amazon GameLift Servers contêineres](#). Crie sua definição de grupo de contêineres no mesmo Região da AWS local em que as imagens do contêiner são armazenadas. Para saber mais sobre como escolher a localização da frota, consulte [Localização da frota](#). No mínimo, faça o seguinte:
 - Crie uma definição de grupo de contêineres de servidores de jogos.

- Adicione uma definição de contêiner com uma imagem de contêiner na versão do seu servidor de jogo.
- Configure um intervalo de portas para os processos do servidor de jogos do contêiner.
- Crie uma frota de contêineres gerenciada. Quando você cria uma frota, começa Amazon GameLift Servers imediatamente a implantar sua versão de servidor de jogos para hospedagem. Você pode configurar vários aspectos de uma frota gerenciada. Para obter orientações, consulte [Crie um Amazon GameLift Servers frota de contêineres gerenciada](#). No mínimo, faça o seguinte:
 - Configure uma função de serviço AWS Identity and Access Management (IAM) para a frota de contêineres. Consulte [Configure uma função de serviço do IAM para Amazon GameLift Servers](#).
 - Especifique a definição do grupo de contêineres do servidor de jogos para implantar nas instâncias da frota.
 - Use valores padrão quando disponíveis para todos os outros parâmetros. Amazon GameLift Servers calcula alguns parâmetros para uma configuração ideal.
- Adicione as frotas de contêineres à sua fila. Em sua fila de sessões de jogo, substitua a frota de teste Anywhere pela sua frota de contêineres gerenciados.
- Teste a hospedagem de jogos com suas frotas de contêineres. Neste ponto, você deve ser capaz de testar toda a solução. Inicie um cliente de jogo e solicite uma sessão de jogo por meio do serviço de back-end. Obtenha informações de conexão e conecte-se a uma sessão de jogo na frota de contêineres.
- Repita as implantações de sua frota. Você pode atualizar as definições de grupos de contêineres e as configurações da frota e, em seguida, implantar atualizações nas frotas existentes.

Etapa 6: personalizar sua solução de contêiner gerenciado

Ao se preparar para o lançamento do jogo, você precisará ajustar seus recursos de hospedagem gerenciada. Algumas das decisões a serem consideradas incluem:

- Otimize a configuração da sua frota de contêineres. Consulte [Personalize uma frota de Amazon GameLift Servers contêineres](#).
- Considere adicionar frotas spot para reduzir custos. Consulte [Tutorial: Crie uma Amazon GameLift Servers fila com instâncias spot](#).
- Se o servidor do jogo precisar comunicar outros AWS recursos, configure funções do IAM para gerenciar o acesso. Consulte [Comunique-se com outros AWS recursos de suas frotas](#).

- Determine onde você deseja posicionar geograficamente os servidores de jogo. Adicione locais remotos às suas frotas gerenciadas. Consulte [Personalize seu Amazon GameLift Servers EC2 frotas gerenciadas](#).
- Experimente as opções de posicionamento de sessões de jogo para frotas gerenciadas, incluindo a personalização das configurações de priorização. Consulte [Personalize uma fila de sessões de jogo](#).
- Configure o ajuste de escala automático da capacidade para atender à demanda esperada dos jogadores. Consulte [Escalaando a capacidade de hospedagem de jogos com Amazon GameLift Servers](#).
- Crie frotas em outras Regiões da AWS e modifique as filas e o escalonamento automático para lidar com os failovers conforme necessário.
- Configure ferramentas de observabilidade de hospedagem, incluindo análises e registros em log. Consulte [Monitoramento Amazon GameLift Servers](#).
- Automatize as implantações de sua frota usando a [infraestrutura como código \(IaC\)](#). Consulte [Gerenciamento Amazon GameLift Servers recursos de hospedagem usando AWS CloudFormation](#).

Amazon GameLift Servers suporta o uso de AWS CloudFormation modelos para qualquer configuração específica da implantação. Você também pode usar o AWS Cloud Development Kit (AWS CDK) para definir seus Amazon GameLift Servers recursos. Para obter mais informações sobre o AWS CDK, consulte o [Guia do AWS Cloud Development Kit \(AWS CDK\) desenvolvedor](#).

Para gerenciar a implantação de suas AWS CloudFormation pilhas, recomendamos o uso de ferramentas e serviços de integração contínua e entrega contínua (CI/CD), como AWS CodePipeline. Essas ferramentas ajudam você a implantar automaticamente ou com aprovação sempre que você cria o binário do servidor de jogos. Com uma CI/CD ferramenta ou serviço, a implantação de recursos para uma nova versão do servidor de jogos pode ter a seguinte aparência:

- Compile e teste o binário do servidor de jogos.
- Faça o upload do binário para Amazon GameLift Servers.
- Implante novas frotas com a nova compilação.
- Adicione as novas frotas à fila de sessões de jogo e remova as frotas com a versão de compilação anterior.
- Quando as frotas com a versão anterior não estiverem mais hospedando sessões de jogo ativas, exclua as AWS CloudFormation pilhas dessas frotas.

Roteiro de desenvolvimento para hospedagem com Anywhere Amazon GameLift Servers

Este roteiro explica como desenvolver uma solução de hospedagem para seu jogo multijogador para usar com seus próprios recursos (hardware local ou máquinas virtuais). Amazon GameLift Servers oferece várias opções de hospedagem de jogos; para obter mais informações sobre essas opções, consulte [Soluções do Amazon GameLift Servers](#).

Com a hospedagem Amazon GameLift Servers Anywhere, seu servidor de jogos é hospedado em recursos de computação que você fornece e gerencia. Você pode criar uma frota Anywhere com as configurações de que precisa e localizada geograficamente onde quer que seus jogadores estejam. Amazon GameLift Servers oferece os seguintes recursos para uma frota Anywhere:

- Gerencia o processo de posicionamento de sessões de jogo para você com base na sua configuração, incluindo:
 - Monitoramento da disponibilidade dos servidores de jogo em suas frotas Anywhere.
 - Processamento de solicitações de jogo do serviço de cliente de jogos e associando essas solicitações de jogos a servidores disponíveis.
 - Solicitação para servidores de jogos das frotas Anywhere iniciarem sessões de jogo.
 - Comunicação de detalhes da conexão aos clientes do jogo.
- Coleta métricas de desempenho para o processo de posicionamento de sessões e métricas de uso para sessões de jogo e jogadores.
- Suporta o conjunto completo de recursos de FlexMatch matchmaking, para que você possa criar um matchmaker e integrá-lo ao sistema de posicionamento da sessão de jogo.
- Oferece ao Amazon GameLift Servers agente a tarefa de lidar com as principais tarefas de gerenciamento de host em uma frota Anywhere.
- Suporta a combinação com frotas Amazon GameLift Servers gerenciadas para uma solução híbrida flexível.

Uma solução Amazon GameLift Servers Anywhere é composta pelos seguintes componentes:

- Uma ou mais frotas Amazon GameLift Servers Anywhere com seus recursos locais ou outros recursos de hospedagem, gerenciados com suas ferramentas existentes de gerenciamento e implantação de configurações. (Opcionalmente, você pode usar AWS Systems Manager o.)

- Um servidor de jogos construído, integrado ao SDK do servidor para ser Amazon GameLift Servers implantado em todas as frotas.
- Um cliente de jogo e serviço de back-end, integrado ao AWS SDK, para interagir com o Amazon GameLift Servers serviço e solicitar sessões de jogo.
- Uma Amazon GameLift Servers fila para colocar novas sessões de jogo com servidores de jogos disponíveis em todas as frotas.
- (Opcional) Um FlexMatch matchmaker para criar partidas multijogador e configurar sessões de jogo para elas.

Este roteiro apresenta um caminho simplificado para colocar seu jogo multijogador em funcionamento com sucesso com a hospedagem Amazon GameLift Servers em qualquer lugar. Depois de configurar os componentes necessários, você pode continuar a iterar no desenvolvimento de jogos e personalizar a solução de hospedagem. À medida que você se aproxima do lançamento, consulte [Preparando seu jogo para o lançamento com Amazon GameLift Servers hospedagem](#) para obter ajuda na preparação da solução de hospedagem para uso em nível de produção.

i Comece com o Amazon GameLift Servers plug-in para Unreal Engine e Unity

Para uma implantação mais rápida, experimente o [Amazon GameLift Servers plug-in](#) para Unreal Engine e Unity. Ele fornece fluxos de trabalho de interface de usuário guiados para implantar rapidamente seu servidor de jogos com configuração mínima, para que você possa testar os componentes do jogo em ação. Depois, você pode aproveitar essa base para criar uma solução de hospedagem personalizada para o seu jogo. Consulte mais detalhes em [Explore com o Amazon GameLift Servers plug-in](#).

Etapa 1: Prepare seu servidor de jogo para trabalhar com Amazon GameLift Servers

Adicione funcionalidade ao seu servidor de jogos para que ele possa se comunicar com o Amazon GameLift Servers serviço quando ele for implantado para hospedagem.

- Obtenha o SDK do servidor para Amazon GameLift Servers (versão 5.x) para seu projeto de jogo. O SDK do servidor está disponível em C++, C# e Go. [Baixe o SDK de um Amazon GameLift Servers servidor](#).

- Modifique o código do servidor de jogos para adicionar a funcionalidade do SDK do servidor. Para obter orientações, consulte [Integre jogos com servidores de jogos personalizados](#). No mínimo, faça o seguinte:
 - Adicione código para inicializar o Amazon GameLift Servers SDK e estabelecer uma WebSocket conexão com o Amazon GameLift Servers serviço. Use a ação `InitSdk()` do SDK do servidor e inclua os parâmetros do servidor, que são necessários para uma frota Anywhere.
 - Adicione código para reportar ao Amazon GameLift Servers serviço quando o processo do servidor estiver pronto para hospedar sessões de jogo. Use a ação `ProcessReady()` do SDK do servidor.
 - Implemente as funções `OnProcessTerminate()` de retorno de chamada necessárias e `OnStartGameSession()`. Com essas funções, os processos do servidor de jogos podem manter uma conexão com o Amazon GameLift Servers serviço, iniciar uma sessão de jogo quando solicitado e responder a uma solicitação para encerrar o processo do servidor de jogos.
- Adicione código para reportar ao Amazon GameLift Servers serviço quando o processo do servidor estiver encerrando uma sessão de jogo. Use a ação `ProcessEnding()` do SDK do servidor.
- Empacote a versão do servidor de jogos. Crie um script de instalação com seus arquivos de compilação, dependências e software de suporte. Consulte [Criar um pacote dos arquivos de compilação do jogo](#). Recomendamos usar um bucket do Amazon Simple Storage Service (Amazon S3) para armazenar as versões da compilação do jogo.
- Teste a integração do servidor de jogos. Para essa tarefa, recomendamos configurar uma frota Amazon GameLift Servers Anywhere para uma estação de trabalho local, conforme descrito em [Configure testes locais com o Amazon GameLift Servers Anywhere](#). Para esta etapa, instale manualmente a compilação do servidor de jogos no dispositivo de teste e inicie um processo no servidor. Use a AWS CLI para solicitar uma nova sessão de jogo e verifique se o Amazon GameLift Servers serviço solicita com êxito que o processo do servidor inicie uma sessão de jogo.

Etapa 2: Prepare o cliente do jogo para entrar em sessões de jogo hospedadas.

Crie uma forma do cliente de jogo solicitar a participação em uma sessão de jogo, obter informações de conexão e conectar-se diretamente a uma sessão de jogo hospedada. A abordagem mais comum é configurar a funcionalidade do serviço de back-end que serve como intermediária entre o cliente do jogo e o serviço. Amazon GameLift Servers Essa abordagem protege os recursos de hospedagem e oferece maior controle sobre como os jogadores são colocados nas sessões de jogo.

- Crie uma funcionalidade de serviço de backend para hospedagem. O serviço de back-end se comunica com o Amazon GameLift Servers serviço e fornece informações de conexão a um cliente de jogo. Essa funcionalidade inclui iniciar sessões de jogo, colocar jogadores em jogos e recuperar informações da sessão de jogo. Para obter orientações, consulte [Integre jogos com servidores de jogos personalizados](#). No mínimo, faça o seguinte:
 - Obtenha o AWS SDK Amazon GameLift Servers e adicione-o ao seu projeto de serviço de back-end. Veja os [recursos Amazon GameLift Servers do SDK para serviços ao cliente](#).
 - Adicione código para inicializar um Amazon GameLift Servers cliente e armazenar as configurações da chave. Consulte [Configurar Amazon GameLift Servers em um serviço de back-end](#).
 - Adicione funcionalidade para chamar a ação do AWS SDK `CreateGameSession()` e fornecer informações de conexão da sessão de jogo a um cliente de jogo. Consulte [Criar uma sessão de jogo em uma frota específica](#).

Ligar `CreateGameSession()` é um ponto de partida conveniente para solicitar novas sessões de jogo. Depois de instalar um sistema de posicionamento de sessões de jogo (consulte a Etapa 3), você substituirá esse código por uma chamada para `StartGameSessionPlacement()` (ou `StartMatchmaking()` se estiver usando `FlexMatch`).

Para obter orientação sobre como projetar o serviço de backend, consulte [Projete seu serviço de cliente de jogos](#).

- Adicione a funcionalidade ao cliente de jogo que permite que os jogadores participem de uma sessão de jogo hospedada. O cliente do jogo faz solicitações para seu serviço de back-end, não diretamente para o Amazon GameLift Servers. Depois que o serviço de backend fornece as informações de conexão da sessão do jogo, o cliente de jogo se conecta diretamente à sessão do jogo para jogar.
- Teste a integração do cliente de jogo. Você pode usar a mesma frota Amazon GameLift Servers Anywhere com estações de trabalho locais para testes.

Etapa 3: Configure o posicionamento da sessão de jogo

Personalize como você deseja Amazon GameLift Servers processar as solicitações de uma nova sessão de jogo e localize os servidores de jogos disponíveis para hospedá-las. Amazon GameLift Servers rastreia automaticamente a disponibilidade de todos os servidores de jogos em todas as frotas. Quando um cliente de jogo envia uma solicitação para participar de uma sessão de jogo,

Amazon GameLift Servers procura o “melhor posicionamento possível” com base em um conjunto de prioridades definidas, como latência, custo e disponibilidade mínimos.

- Crie uma fila de sessões de jogo para alocar uma nova sessão de jogo em servidores de jogos disponíveis. As filas são o principal mecanismo para posicionamento de sessões de jogo. Para obter orientações, consulte [Criar uma fila de sessões de jogo](#).
- No mínimo, adicione suas frotas Anywhere como destinos na fila. Todas as outras configurações são opcionais.
- No código de serviço de backend, converta a chamada **CreateGameSession()** em **StartGameSessionPlacement()**. Consulte [Criar uma sessão de jogo em uma fila com vários locais](#).
- Crie um mecanismo para notificar um cliente de jogo quando uma sessão de jogo estiver pronta para ser iniciada. Durante o desenvolvimento, você pode pesquisar o status da sessão do jogo usando uma chamada para DescribeGameSessionPlacement. No entanto, antes de usar uma fila para processar grandes volumes, você precisará ativar as notificações de eventos. Consulte [Configurar notificações de eventos para o posicionamento de sessões de jogo](#).
- (Opcional) Adicione componentes de FlexMatch matchmaking. Para obter orientação, consulte o [guia do Amazon GameLift ServersFlexMatch desenvolvedor](#).

Etapa 4: configurar uma frota em qualquer lugar com o Amazon GameLift Servers agente

Até agora, você estava trabalhando com dispositivos locais (registrados como Anywhere fleet computes) para testar e iterar os componentes do seu jogo. A próxima etapa é configurar o tipo de frota necessária para um sistema de produção. Para esses recursos, use o Amazon GameLift Servers Agente para gerenciar algumas das principais tarefas de gerenciamento do host no computador. Consulte mais detalhes em [Trabalhe com o Amazon GameLift Servers agente](#).

- Obtenha o Amazon GameLift Servers Agente e adicione-o ao pacote de instalação do servidor do jogo. Obtenha e crie o código-fonte do Agent, disponível no [repositório do Amazon GameLift Servers Agent Github](#). Coloque o arquivo executável JAR resultante no mesmo diretório do executável de compilação do jogo.
- Modifique o script de inicialização para o Agent conforme necessário. Certifique-se de que o executável do Agent seja iniciado assim que a computação começar a ser executada. Consulte o arquivo leia-me no repositório do Agent para obter ajuda com a instalação e a execução do Agent nos recursos de computação de hospedagem. Seu comando de inicialização deve incluir opções

para especificar, no mínimo, o ID e a Região da AWS da frota Anywhere, um local personalizado e um nome de computação.

O Agent gerencia automaticamente as seguintes tarefas para você. Portanto, se estiver lidando com essas tarefas com scripts, você poderá removê-los:

- Chama `RegisterCompute()` para adicionar a computação a uma frota Anywhere.
- Chamadas `GetComputeAuthToken()` para autenticar servidores de jogos quando eles se conectam ao Amazon GameLift Servers serviço. O Agent gerencia a obtenção e a atualização do token de autenticação, que pode ser usado por todos os processos do servidor de jogos em execução na computação.
- Inicia novos processos do servidor na computação com base em um conjunto de instruções de tempo de execução.
- Crie uma configuração de tempo de execução para computações na frota Anywhere. Você pode usar o Amazon GameLift Servers console ou a AWS CLI para criar ou modificar instruções de tempo de execução para a frota. O Agente executa essas instruções e solicita periodicamente atualizações do Amazon GameLift Servers serviço.
- Configure ou modifique a fila de sessões de jogo conforme necessário. Crie uma nova fila (ou atualize uma existente) para usar as frotas Anywhere que são implantadas com o Agente. Amazon GameLift Servers
- Teste a integração do Agent com suas frotas Anywhere. Verifique se o Agent está iniciando corretamente os processos do servidor com base na configuração de tempo de execução.

Etapa 5: Personalize suas frotas Anywhere

Ao se preparar para o lançamento do jogo, você precisará ajustar seus recursos de hospedagem gerenciada. Algumas das decisões a serem consideradas incluem:

- Automatize o processo de iniciar e encerrar os recursos de computação conforme necessário, incluindo a instalação e a execução do software do servidor de jogos. Reciclar os recursos de computação é útil para garantir que sejam atualizadas regularmente, e encerrá-las pode economizar custos quando não forem necessárias.
- Se o servidor do jogo precisar comunicar outros AWS recursos, configure funções do IAM para gerenciar o acesso. Consulte [Comunique-se com outros AWS recursos de suas frotas](#).

- Determine onde você deseja posicionar geograficamente os servidores de jogo. Adicione locais remotos às suas frotas gerenciadas. Consulte [Personalize seu Amazon GameLift Servers EC2 frotas gerenciadas](#).
- Otimize o desempenho da frota selecionando configurações de recursos computacionais e, em seguida, configure as instruções de tempo de execução para executar um número ideal de processos de servidor por computação.
- Experimente as opções de posicionamento de sessões de jogo para frotas gerenciadas, incluindo a personalização das configurações de priorização. Consulte [Personalize uma fila de sessões de jogo](#).
- Crie mecanismos para lidar com a escalabilidade de capacidade manual ou automatizada para atender à demanda esperada dos jogadores. Considere quais fatores devem levar o sistema a aumentar ou diminuir o número de recursos de computação disponíveis para hospedar sessões de jogo.
- Projete e implemente o failover para outros recursos, se necessário.
- Configure ferramentas de observabilidade de hospedagem, incluindo análises e registros em log. Consulte [Monitoramento Amazon GameLift Servers](#).

Roteiro de desenvolvimento para hospedagem híbrida com Amazon GameLift Servers

Este roteiro orienta você sobre como desenvolver uma solução de hospedagem para seu jogo multijogador. Amazon GameLift Servers oferece várias opções de hospedagem de jogos; para obter mais informações sobre essas opções, consulte [Soluções do Amazon GameLift Servers](#).

Uma solução híbrida usa uma combinação de recursos de hospedagem, incluindo recursos baseados em nuvem gerenciados por Amazon GameLift Servers e seus próprios recursos de hospedagem autogerenciados. Para uma discussão mais detalhada sobre hospedagem híbrida, consulte este artigo: [Hospedagem híbrida de servidores de jogos com Amazon GameLift Servers Anywhere](#). Com Amazon GameLift Servers, você pode configurar uma solução híbrida que usa componentes e processos comuns, para que você possa gerenciar centralmente uma frota global e mover cargas com facilidade entre todos os tipos de recursos.

Uma arquitetura híbrida consiste nos seguintes componentes:

- Uma ou mais frotas Amazon GameLift Servers gerenciadas, que usam instâncias do Amazon Elastic Compute Cloud EC2 (Amazon) otimizadas para hospedagem de jogos multijogador.

- Uma ou mais frotas Amazon GameLift Servers Anywhere, que usam seus recursos locais existentes ou outros recursos de hospedagem, incluindo seu gerenciamento de configuração e ferramentas de implantação. (Opcionalmente, você pode usar o AWS Systems Manager.)
- Um único servidor de jogo construído, integrado ao SDK do servidor Amazon GameLift Servers, para ser implantado em todas as frotas.
- Um único cliente de jogo e serviço de back-end, integrado ao AWS SDK, para interagir com o Amazon GameLift Servers serviço e solicitar sessões de jogo.
- Uma Amazon GameLift Servers fila compartilhada para colocar novas sessões de jogo com os servidores de jogos disponíveis e equilibrar a carga em todas as frotas.
- O Amazon GameLift Servers Agent, que é implantado com uma frota Anywhere, para simplificar as tarefas de gerenciamento de processos do servidor em todos os computadores em todas as frotas.
- (Opcional) Um FlexMatch matchmaker para criar partidas multijogador e configurar sessões de jogo para elas.

Este roteiro apresenta um caminho simplificado para colocar seu jogo multijogador em funcionamento com sucesso em uma solução de hospedagem híbrida com Amazon GameLift Servers. Depois de configurar os componentes necessários, você pode continuar a iterar no desenvolvimento de jogos e personalizar a solução de hospedagem. À medida que você se aproxima do lançamento, consulte [Preparando seu jogo para o lançamento com Amazon GameLift Servers hospedagem](#) para obter ajuda na preparação da solução de hospedagem para uso em nível de produção.

Comece com o Amazon GameLift Servers plugin

Se você estiver desenvolvendo projetos com o Unreal Engine ou o Unity, comece a configurar seu jogo para hospedagem com o Amazon GameLift Servers plug-in. Com o plug-in, você pode adicionar o Amazon GameLift Servers SDKs ao seu projeto de jogo e usar os fluxos de trabalho guiados para criar uma versão funcional simples de uma solução de hospedagem híbrida com uma frota Anywhere e uma frota Amazon GameLift Servers gerenciada. Em seguida, você pode usar esses fundamentos para compilar e personalizar conforme necessário.

Etapa 1: Prepare seu servidor de jogo para trabalhar com Amazon GameLift Servers

Adicione funcionalidade ao seu servidor de jogos para que ele possa se comunicar com o Amazon GameLift Servers serviço quando ele for implantado para hospedagem. A mesma funcionalidade é necessária se o servidor do jogo estiver sendo executado em uma frota Amazon GameLift Servers gerenciada ou em uma frota Anywhere.

- Obtenha o SDK do servidor para Amazon GameLift Servers (versão 5.x) para seu projeto de jogo. O SDK do servidor está disponível em C++, C# e Go. [Baixe o SDK do servidor para Amazon GameLift Servers](#).
- Modifique o código do servidor de jogos para adicionar a funcionalidade do SDK do servidor. Para obter orientações, consulte [Integre jogos com servidores de jogos personalizados](#). No mínimo, faça o seguinte:
 - Adicione código para inicializar o Amazon GameLift Servers SDK e estabelecer uma WebSocket conexão com o Amazon GameLift Servers serviço. Use a ação `InitSdk()` do SDK do servidor. Inclua código para especificar os parâmetros do servidor ao executar em uma computação de frota Anywhere.
 - Adicione código para reportar ao Amazon GameLift Servers serviço quando o processo do servidor estiver pronto para hospedar sessões de jogo. Use a ação `ProcessReady()` do SDK do servidor.
 - Implemente as funções `OnProcessTerminate()` de retorno de chamada necessárias e `OnStartGameSession()`. Com essas funções, os processos do servidor de jogos podem manter uma conexão com o Amazon GameLift Servers serviço, iniciar uma sessão de jogo quando solicitado e responder a uma solicitação para encerrar o processo do servidor de jogos. Amazon GameLift Servers
 - Adicione código para reportar ao Amazon GameLift Servers serviço quando o processo do servidor estiver encerrando uma sessão de jogo. Use a ação `ProcessEnding()` do SDK do servidor.
- Empacote a versão do servidor de jogos. Crie um script de instalação com seus arquivos de compilação, dependências e software de suporte. Consulte [Criar um pacote dos arquivos de compilação do jogo](#). Recomendamos usar um bucket do Amazon Simple Storage Service (Amazon S3) para armazenar as versões da compilação do jogo.
- Teste a integração do servidor de jogos. Para essa tarefa, recomendamos configurar uma frota Amazon GameLift Servers Anywhere com uma estação de trabalho local, conforme descrito em [Configure testes locais com o Amazon GameLift Servers Anywhere](#). Para esta etapa, instale manualmente a compilação do servidor de jogos no dispositivo de teste e inicie um processo

no servidor. Use a AWS CLI para solicitar uma nova sessão de jogo e verifique se o Amazon GameLift Servers serviço solicita com êxito que o processo do servidor inicie uma sessão de jogo.

Etapa 2: Prepare o cliente do jogo para entrar em sessões de jogo hospedadas.

Crie uma forma do cliente de jogo solicitar a participação em uma sessão de jogo, obter informações de conexão e conectar-se diretamente a uma sessão de jogo hospedada. A abordagem mais comum é configurar a funcionalidade do serviço de back-end que serve como intermediária entre o cliente do jogo e o serviço. Amazon GameLift Servers Essa abordagem protege os recursos de hospedagem e oferece maior controle sobre como os jogadores são colocados nas sessões de jogo.

- Crie uma funcionalidade de serviço de backend para hospedagem. O serviço de back-end se comunica com o Amazon GameLift Servers serviço e fornece informações de conexão a um cliente de jogo. Essa funcionalidade inclui iniciar sessões de jogo, colocar jogadores em jogos e recuperar informações da sessão de jogo. Para obter orientações, consulte [Integre jogos com servidores de jogos personalizados](#). No mínimo, faça o seguinte:
 - Obtenha o AWS SDK Amazon GameLift Servers e adicione-o ao seu projeto de serviço de back-end. Veja os [recursos Amazon GameLift Servers do SDK para serviços ao cliente](#).
 - Adicione código para inicializar um Amazon GameLift Servers cliente e armazenar as configurações da chave. Consulte [Configurar Amazon GameLift Servers em um serviço de back-end](#).
 - Adicione funcionalidade para chamar a ação do AWS SDK `CreateGameSession()` e fornecer informações de conexão da sessão de jogo a um cliente de jogo. Consulte [Criar uma sessão de jogo em uma frota específica](#).

Ligar `CreateGameSession()` é um ponto de partida conveniente para solicitar novas sessões de jogo. Depois de instalar um sistema de posicionamento de sessões de jogo (consulte a Etapa 3), você substituirá esse código por uma chamada para `StartGameSessionPlacement()` (ou `StartMatchmaking()` se estiver usando `FlexMatch`).

Para obter orientação sobre como projetar o serviço de backend, consulte [Projete seu serviço de cliente de jogos](#).

- Adicione a funcionalidade ao cliente de jogo que permite que os jogadores participem de uma sessão de jogo hospedada. O cliente do jogo faz solicitações para seu serviço de back-end, não diretamente para o. Amazon GameLift Servers Depois que o serviço de backend fornece as informações de conexão da sessão do jogo, o cliente de jogo se conecta diretamente à sessão do jogo para jogar.

- Teste a integração do cliente de jogo. Você pode usar a mesma frota Amazon GameLift Servers Anywhere com uma estação de trabalho local para testes.

Durante a fase de desenvolvimento, se você quiser testar como sua compilação de jogos se comporta em uma frota Amazon GameLift Servers gerenciada, recomendamos que você também configure um ambiente de teste baseado em [nuvem](#). Essa solução Amazon GameLift Servers de kit de ferramentas imita o comportamento de uma frota gerenciada, mas permite que você atualize as compilações do servidor de jogos com o mínimo de tempo de resposta.

Etapa 3: Configure o posicionamento da sessão de jogo

Personalize como você deseja Amazon GameLift Servers processar as solicitações de uma nova sessão de jogo e localize os servidores de jogos disponíveis para hospedá-las. Amazon GameLift Servers rastreia automaticamente a disponibilidade de todos os servidores de jogos em todas as frotas. Quando um cliente de jogo envia uma solicitação para participar de uma sessão de jogo, Amazon GameLift Servers procura o “melhor posicionamento possível” com base em um conjunto de prioridades definidas, como latência, custo e disponibilidade mínimos.

- Crie uma fila de sessões de jogo para alocar uma nova sessão de jogo em servidores de jogos disponíveis. As filas são o principal mecanismo para posicionamento de sessões de jogo. Para obter orientações, consulte [Criar uma fila de sessões de jogo](#).
- No mínimo, adicione suas frotas Anywhere como destinos na fila. Todas as outras configurações são personalizações opcionais.
- No código de serviço de backend, converta a chamada **CreateGameSession()** em **StartGameSessionPlacement()**. Consulte [Criar uma sessão de jogo em uma fila com vários locais](#).
- Crie um mecanismo para notificar um cliente de jogo quando uma sessão de jogo estiver pronta para ser iniciada. Durante o desenvolvimento, você pode pesquisar o status da sessão do jogo usando uma chamada para `DescribeGameSessionPlacement`. No entanto, antes de usar uma fila para processar grandes volumes, você precisará ativar as notificações de eventos. Consulte [Configurar notificações de eventos para o posicionamento de sessões de jogo](#).
- (Opcional) Adicione componentes de FlexMatch matchmaking. Para obter orientação, consulte o [guia do Amazon GameLift Servers FlexMatch desenvolvedor](#).

Etapa 4: configurar uma frota em qualquer lugar com o Amazon GameLift Servers agente

Até agora, você estava trabalhando com dispositivos locais (registrados como Anywhere fleet computes) para testar e iterar os componentes do seu jogo. A próxima etapa é configurar o tipo de frota que você precisará para um sistema de produção. Comece com uma frota Anywhere e adicione o Amazon GameLift Servers Agente para gerenciar algumas das principais tarefas de gerenciamento de host no computador. Consulte mais detalhes em [Trabalhe com o Amazon GameLift Servers agente](#).

- Obtenha o Amazon GameLift Servers Agente e adicione-o ao pacote de instalação do servidor do jogo. Obtenha e crie o código-fonte do Agent, disponível no [repositório do Amazon GameLift Servers Agent Github](#). Coloque o arquivo executável JAR resultante no mesmo diretório do executável de compilação do jogo.
- Modifique o script de inicialização para o Agent conforme necessário. Certifique-se de que o executável do Agent seja iniciado assim que a computação começar a ser executada. Consulte o arquivo leia-me no repositório do Agent para obter ajuda com a instalação e a execução do Agent nos recursos de computação de hospedagem. Seu comando de inicialização deve incluir opções para especificar, no mínimo, o ID da frota Anywhere Região da AWS, um local personalizado e um nome de computação.

O Agent gerencia automaticamente as seguintes tarefas para você. Portanto, se estiver lidando com essas tarefas com scripts, você poderá removê-los:

- Chama `RegisterCompute()` para adicionar a computação a uma frota Anywhere.
- Chamadas `GetComputeAuthToken()` para autenticar servidores de jogos quando eles se conectam ao Amazon GameLift Servers serviço. O Agent gerencia a obtenção e a atualização do token de autenticação, que pode ser usado por todos os processos do servidor de jogos em execução na computação.
- Inicia novos processos do servidor na computação com base em um conjunto de instruções de tempo de execução.
- Crie uma configuração de tempo de execução para computações na frota Anywhere. No mínimo, especifique o caminho de inicialização para o executável do servidor de jogos. Você pode usar o Amazon GameLift Servers console ou a AWS CLI para criar ou modificar instruções de tempo de execução para a frota. O Agente executa essas instruções e solicita periodicamente atualizações do Amazon GameLift Servers serviço.

- Configure ou modifique a fila de sessões de jogo conforme necessário. Crie uma nova fila (ou atualize uma existente) e designe um destino para a frota Anywhere implantada com o Agente. Amazon GameLift Servers
- Teste a integração do Agent com suas frotas Anywhere. Verifique se o Agent está iniciando corretamente os processos do servidor com base na configuração de tempo de execução.

Etapa 5: Crie uma frota gerenciada baseada na nuvem

Crie uma EC2 frota Amazon GameLift Servers gerenciada para complementar sua frota Anywhere. Se você configurou um ambiente de teste baseado na nuvem na Etapa 2 para acelerar o desenvolvimento, planeje criar uma frota gerenciada depois de concluir a maior parte do desenvolvimento e teste do jogo. Você precisa de uma frota totalmente gerenciada para definir e testar configurações adicionais, como a escalabilidade automática da capacidade.

- Package a compilação do seu servidor de jogo e faça o upload para Amazon GameLift Servers. Crie um script de instalação com seus arquivos de compilação, dependências e software de suporte. Você pode usar o mesmo software de compilação com as frotas Anywhere e gerenciadas. Consulte [Implemente uma criação de servidor personalizada para Amazon GameLift Servers hospedagem](#). Você pode fazer o upload da sua compilação Amazon GameLift Servers usando o console ou a AWS CLI.

Antes de enviar a compilação, decida em que Região da AWS você deseja criar a frota gerenciada. É necessário fazer o upload da compilação para a mesma região. Para saber mais sobre como escolher a localização da frota, consulte [Localização da frota](#).

- Crie uma EC2 frota gerenciada. Você pode usar o Amazon GameLift Servers console ou a AWS CLI para criar uma frota gerenciada. Quando você cria uma frota, começa Amazon GameLift Servers imediatamente a implantar sua versão de servidor de jogos para hospedagem. Você pode configurar vários aspectos de uma frota gerenciada. Para obter orientações, consulte [Crie uma EC2 frota Amazon GameLift Servers gerenciada](#). No mínimo, faça o seguinte:
 - Dê um nome à frota e especifique qual compilação de jogo enviada deve ser implantada.
 - Escolha instâncias sob demanda para a frota e selecione um tipo de instância que esteja disponível na localização da frota. As frotas spot são uma opção valiosa, mas exigem design e configuração adicionais.
 - Crie uma configuração de tempo de execução com configurações semelhantes às usadas com a frota Anywhere. No mínimo, especifique o caminho de inicialização para o executável do servidor de jogos.

- Especifique as configurações de porta para permitir que o tráfego de entrada acesse os servidores de jogos.
- Adicione a frota gerenciada à fila de sessões de jogo compartilhadas. Atualize a fila da Etapa 4 para que ela inclua destinos tanto para a frota gerenciada quanto para a frota Anywhere implantada com o Amazon GameLift Servers Agente.
- Teste a hospedagem de jogos com as frotas gerenciadas. Nesse ponto, você poderá testar todo o ciclo de hospedagem, com um cliente de jogo solicitando uma sessão de jogo, obtendo informações de conexão e se conectando com sucesso a uma sessão de jogo.

Etapa 6: Personalize as frotas

Ao se preparar para o lançamento do jogo, você precisará ajustar as soluções de hospedagem. Algumas das decisões a serem consideradas incluem:

- Para frotas Anywhere, automatize o processo de iniciar e desligar recursos de computação conforme necessário, incluindo a instalação e a execução do software de servidor de jogos. Reciclar as computações é útil para garantir que sejam atualizadas regularmente, e encerrá-las pode economizar custos quando não forem necessárias.
- Se o servidor do jogo precisar comunicar outros recursos da AWS, configure os perfis do IAM para gerenciar o acesso. Consulte [Comunique-se com outros AWS recursos de suas frotas](#).
- Determine onde você deseja posicionar geograficamente os servidores de jogo. Adicione locais remotos às suas frotas gerenciadas. Consulte [Personalize seu Amazon GameLift Servers EC2 frotas gerenciadas](#).
- Para frotas gerenciadas, considere usar frotas Spot para reduzir custos. Consulte [Tutorial: Crie uma Amazon GameLift Servers fila com instâncias spot](#).
- Otimize o desempenho da frota selecionando configurações de recursos de computação e configure as instruções de tempo de execução para executar o número ideal de processos de servidor por computação. Faça isso tanto para frotas Anywhere quanto para frotas gerenciadas. Consulte [Gerencie como Amazon GameLift Servers lança servidores de jogos](#).
- Experimente as opções de posicionamento de sessões de jogo para frotas gerenciadas, incluindo a personalização das configurações de priorização. Consulte [Personalize uma fila de sessões de jogo](#).
- Para frotas gerenciadas, configure a escalabilidade automática da capacidade para atender à demanda esperada dos jogadores. Consulte [Escalando a capacidade de hospedagem de jogos com Amazon GameLift Servers](#).

- Para frotas Anywhere, crie mecanismos para lidar com a escalabilidade de capacidade manual ou automatizada para atender à demanda esperada dos jogadores.
- Projete e implemente o failover para outros recursos, se necessário. Configure frotas em espera em outras Regiões da AWS e modifique as filas e o ajuste de escala automático para lidar com failovers, se necessário.
- Configure ferramentas de observabilidade de hospedagem, incluindo análises e registros em log. Consulte [Monitoramento Amazon GameLift Servers](#). Crie grupos de métricas para agregar analytics para todos os recursos de hospedagem.
- Automatize a implantação usando [infraestrutura como código \(IaC\)](#). Consulte [Gerenciamento Amazon GameLift Servers recursos de hospedagem usando AWS CloudFormation](#).

Amazon GameLift Servers suporta o uso de AWS CloudFormation modelos para qualquer configuração específica da implantação. Você também pode usar o AWS Cloud Development Kit (AWS CDK) para definir seus Amazon GameLift Servers recursos. Para obter mais informações sobre o AWS CDK, consulte o [Guia do AWS Cloud Development Kit \(AWS CDK\) desenvolvedor](#).

Para gerenciar a implantação de suas AWS CloudFormation pilhas, recomendamos o uso de ferramentas e serviços de integração contínua e entrega contínua (CI/CD), como AWS CodePipeline. Essas ferramentas ajudam você a implantar automaticamente ou com aprovação sempre que você cria o binário do servidor de jogos. Com uma CI/CD ferramenta ou serviço, a implantação de recursos para uma nova versão do servidor de jogos pode ter a seguinte aparência:

- Compile e teste o binário do servidor de jogos.
- Faça o upload do binário para Amazon GameLift Servers.
- Implante novas frotas com a nova compilação.
- Adicione as novas frotas à fila de sessões de jogo e remova as frotas com a versão de compilação anterior.
- Quando as frotas com a compilação anterior não estiverem mais hospedando sessões de jogo ativas, exclua as pilhas do AWS CloudFormation dessas frotas.

Preparando jogos para Amazon GameLift Servers

Prepare seus jogos multijogador para serem hospedados no Amazon GameLift Servers. Integrar Amazon GameLift Servers hospede recursos em seus projetos de jogos e construa seu servidor de jogos e servidores de clientes. Configure um ambiente de teste hospedado para oferecer suporte ao desenvolvimento e testes iterativos rápidos do jogo.

Tópicos

- [Integre jogos com servidores de jogos personalizados](#)
- [Projete seu serviço de cliente de jogos](#)
- [Configure para desenvolvimento iterativo com Amazon GameLift Servers Anywhere](#)
- [Teste sua integração usando Amazon GameLift Servers Local](#)
- [Adicionando FlexMatch matchmaking](#)
- [Obtenha dados da frota para uma Amazon GameLift Servers instância](#)
- [Integrando jogos com Amazon GameLift ServersRealtime](#)

Integre jogos com servidores de jogos personalizados

Amazon GameLift Servers fornece um conjunto completo de ferramentas para preparar seus jogos multijogador e servidores de jogos personalizados para serem executados Amazon GameLift Servers. O Amazon GameLift Servers SDKs contêm bibliotecas necessárias para que clientes e servidores de jogos se comuniquem com Amazon GameLift Servers. Para obter mais informações sobre SDKs e onde obtê-los, consulte [Obtenha ferramentas Amazon GameLift Servers de desenvolvimento](#).

Os tópicos desta seção contêm instruções detalhadas sobre como adicionar Amazon GameLift Servers funcionalidade para seu cliente de jogo e servidor de jogos antes da implantação em Amazon GameLift Servers.

Tópicos

- [Interações cliente/servidor do jogo com Amazon GameLift Servers](#)
- [Integre seu servidor de jogos com Amazon GameLift Servers](#)
- [Integre seu cliente de jogo com Amazon GameLift Servers](#)
- [Motores de jogo e Amazon GameLift Servers](#)

Interações cliente/servidor do jogo com Amazon GameLift Servers

Os componentes do seu Amazon GameLift Servers as soluções de hospedagem interagem umas com as outras de maneiras específicas para realizar sessões de jogo em resposta à demanda do jogador. Este tópico descreve como os componentes se comunicam entre si quando o servidor do jogo está hospedado no Amazon GameLift Servers EC2 frotas gerenciadas, autogerenciadas Amazon GameLift Servers Frotas em qualquer lugar ou uma solução híbrida.

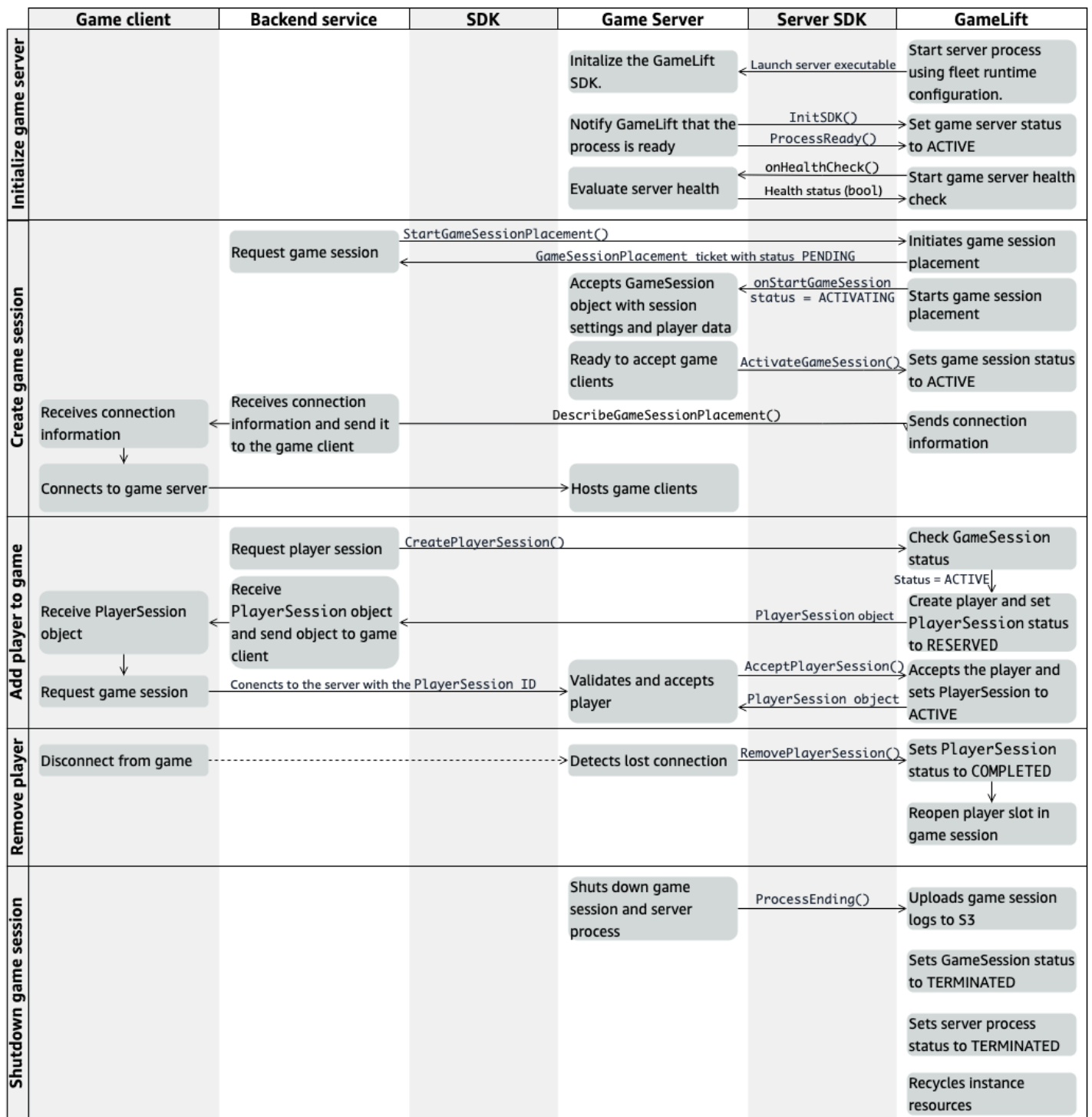
Os componentes da solução de hospedagem incluem um servidor de jogos, o Amazon GameLift Servers serviço, um serviço de back-end do lado do cliente e um cliente de jogo. O servidor do jogo usa o [Amazon GameLift Servers SDK do servidor](#) para interagir com o Amazon GameLift Servers serviço. O serviço de back-end usa o [Amazon GameLift Servers API de serviço](#) (parte do AWS SDK) para interagir com o serviço em nome do cliente do jogo. Ao entrar em uma sessão de jogo, o cliente do jogo se conecta diretamente a uma sessão de jogo usando o endereço IP e o número da porta exclusivos da sessão de jogo.

Tópicos

- [Diagrama de interações](#)
- [Comportamentos de interação](#)

Diagrama de interações

O diagrama a seguir ilustra como seus componentes de hospedagem de jogos interagem para que o Amazon GameLift Servers o serviço pode rastrear o status da disponibilidade do servidor de jogos e iniciar sessões de jogo em resposta às demandas dos jogadores.



Comportamentos de interação

As seções a seguir descrevem a sequência de eventos em cada uma das principais interações.

Inicializar um processo do servidor de jogos

Na inicialização, um processo de servidor de jogos estabelece comunicação com o Amazon GameLift Servers serviço e relata seu status como pronto para hospedar uma sessão de jogo.

1. Um novo processo do executável do servidor do jogo começa a ser executado em um recurso de hospedagem.
2. O processo do servidor de jogos chama as seguintes operações do SDK do servidor em sequência:
 - a. `InitSDK()` para inicializar o SDK do servidor, autenticar o processo do servidor e estabelecer comunicação com o Amazon GameLift Servers serviço.
 - b. `ProcessReady()` para comunicar a prontidão para hospedar uma sessão de jogo. Essa chamada também relata as informações de conexão do processo, que os clientes do jogo usam para se conectar à sessão do jogo e outras informações.

O processo do servidor então aguarda as solicitações do Amazon GameLift Servers serviço.

3. Amazon GameLift Servers atualiza o status do processo do servidor para `ACTIVE` e está disponível para hospedar uma nova sessão de jogo.
4. Amazon GameLift Servers começa a chamar periodicamente o `onHealthCheck` retorno de chamada para solicitar um status de saúde dos processos do servidor. Essas chamadas continuam enquanto o processo do servidor permanece ativo. O processo do servidor deve responder como íntegro ou não dentro de um minuto. Se o processo do servidor não responder corretamente ou não responder, em algum momento o Amazon GameLift Servers O serviço altera o status ativo do processo do servidor e interrompe o envio de solicitações para iniciar a sessão do jogo.

Criar uma sessão de jogo

A ferramenta Amazon GameLift Servers o serviço inicia uma nova sessão de jogo em resposta a uma solicitação de um jogador para jogar o jogo.

1. Um jogador utilizando o cliente do jogo solicita ingressar em uma sessão de jogo. Dependendo de como o jogo lida com o processo de ingresso de jogadores, o cliente do jogo envia uma solicitação ao serviço de backend.
2. Se o processo de ingresso do jogador exigir o início de uma nova sessão de jogo, o serviço de back-end enviará uma solicitação para uma nova sessão de jogo para o Amazon GameLift

Servers serviço. Essa solicitação chama a operação `StartGameSessionPlacement()` da API de serviço. (Como alternativa, o serviço de backend pode chamar `StartMatchmaking()` ou `CreateGameSession()`.)

3. A ferramenta Amazon GameLift Servers o serviço responde criando um novo `GameSessionPlacement` ticket com status `PENDING`. Ele retorna as informações do tíquete ao serviço de backend, para que ele possa acompanhar o status do tíquete de posicionamento e determinar quando a sessão do jogo está pronta para os jogadores. Para obter mais informações, consulte [Configurar notificações de eventos para o posicionamento de sessões de jogo](#).
4. A ferramenta Amazon GameLift Servers o serviço inicia o processo de colocação da sessão de jogo. Ele identifica quais frotas examinar e pesquisa essas frotas em busca de um processo de servidor ativo que não esteja hospedando uma sessão de jogo. Ao localizar um processo de servidor disponível, o Amazon GameLift Servers o serviço faz o seguinte:
 - a. Cria um objeto `GameSession` com as configurações da sessão de jogo e os dados do jogador da solicitação de posicionamento e define o status como `ACTIVATING`.
 - b. Solicita que o processo do servidor inicie uma sessão de jogo. O serviço invoca o retorno de chamada `onStartGameSession` do processo de servidor e transmite o objeto `GameSession`.
 - c. Altera o número de sessões de jogo do processo do servidor para 1.
5. O processo do servidor executa sua função de retorno de chamada `onStartGameSession`. Quando o processo do servidor estiver pronto para aceitar conexões de jogadores, ele chama a operação `ActivateGameSession()` do SDK do servidor e aguarda por conexões de jogadores.
6. A ferramenta Amazon GameLift Servers O serviço atualiza o `GameSession` objeto com as informações de conexão do processo do servidor (conforme relatado na chamada `paraProcessReady()`) e define o status da sessão do jogo como `ACTIVE`. Ele também atualiza o status do tíquete `GameSessionPlacement` para `FULFILLED`.
7. O serviço de backend chama `DescribeGameSessionPlacement()` para verificar o status do tíquete e obter informações sobre a sessão do jogo. Quando a sessão do jogo está ativa, o serviço de backend notifica o cliente do jogo e transmite as informações de conexão da sessão do jogo.
8. O cliente do jogo usa as informações de conexão para se conectar diretamente ao processo do servidor de jogo e ingressar na sessão de jogo.

Adição de um jogador a um jogo

Opcionalmente, um jogo pode usar sessões de jogadores para acompanhar as conexões dos jogadores às sessões de jogo. As sessões de jogadores podem ser criadas individualmente ou como parte de uma solicitação de posicionamento de sessão de jogo.

1. O serviço de backend chama a operação `CreatePlayerSession()` da API do serviço com um ID de sessão do jogo.
2. A ferramenta Amazon GameLift Servers O serviço verifica o status da sessão de jogo (deve ser `ACTIVE`) e procura uma vaga de jogador aberta na sessão de jogo. Se o slot estiver disponível, o serviço fará o seguinte:
 - a. Cria um objeto `PlayerSession` e define o status como `RESERVED`.
 - b. Responde à solicitação de serviço de backend com as informações da sessão do jogador.
3. O serviço de backend transmite as informações da sessão do jogador para o cliente do jogo junto com as informações de conexão da sessão do jogo.
4. O cliente do jogo usa as informações de conexão e o ID da sessão de jogador para se conectar diretamente ao processo do servidor de jogo e solicitar o ingresso na sessão de jogo.
5. Em resposta a uma tentativa de ingresso do cliente de jogo, o processo do servidor de jogos chama a operação `AcceptPlayerSession()` da API do serviço para validar o ID da sessão do jogador. O processo do servidor aceita ou rejeita a conexão.
6. A ferramenta Amazon GameLift Servers O serviço executa uma das seguintes ações:
 - a. Se a conexão for aceita, então Amazon GameLift Servers define o `PlayerSession` status `ACTIVE` e o passa `PlayerSession` para o processo do servidor do jogo.
 - b. Se o processo do servidor do jogo não exigir `AcceptPlayerSession()` o ID da sessão do jogador dentro de um determinado período de tempo após a `CreatePlayerSession()` solicitação original, o Amazon GameLift Servers O serviço altera o `PlayerSession` status `TIMEDOUT` e reabre o espaço do jogador na sessão de jogo.

Remoção de um jogador

Para jogos que usam sessões de jogadores, o processo do servidor de jogos notifica o Amazon GameLift Servers serviço quando um jogador se desconecta. O serviço usa essas informações para acompanhar o status dos slots dos jogadores em uma sessão de jogo e pode permitir que novos jogadores usem os slots abertos.

1. Um jogador se desconecta da sessão do jogo.
2. O processo do servidor de jogo detecta a conexão perdida e chama a operação `RemovePlayerSession()` do SDK do servidor.
3. A ferramenta Amazon GameLift Servers O serviço altera o status da sessão do jogador para `COMPLETED` e reabre o espaço do jogador na sessão de jogo.

Encerramento da sessão de jogo

No final de uma sessão de jogo, ou ao encerrar a sessão de jogo, o processo do servidor notifica o Amazon GameLift Servers serviço de status da sessão de jogo.

1. O processo do servidor de jogos encerra a sessão do jogo e inicia o encerramento do processo chamando a operação `ProcessEnding()` do SDK do servidor.
2. A ferramenta Amazon GameLift Servers o serviço faz o seguinte:
 - a. Carrega os logs de sessão de jogo no Amazon Simple Storage Service (Amazon S3).
 - b. Altera o status da sessão do jogo para `TERMINATED`.
 - c. Altera o status do processo do servidor para `TERMINATED`.
 - d. Dependendo de como a solução de hospedagem é projetada, os recursos de hospedagem recém-disponíveis são alocados para executar um novo processo de servidor de jogos.

Integre seu servidor de jogos com Amazon GameLift Servers

Depois que seu servidor de jogo personalizado for implantado e executado em Amazon GameLift Servers instâncias, ele deve ser capaz de interagir com Amazon GameLift Servers (e potencialmente outros recursos). Esta seção descreve como integrar seu software de servidor de jogos com Amazon GameLift Servers.

Note

Essas instruções pressupõem que você criou um Conta da AWS e que tem um projeto de servidor de jogo existente.

Os tópicos nesta seção descrevem como lidar com as seguintes tarefas de integração:

- Estabelecer comunicação entre Amazon GameLift Servers e seus servidores de jogos.

- Gere e use um certificado TLS para estabelecer uma conexão segura entre o cliente do jogo e o servidor do jogo.
- Forneça permissões para que seu software de servidor de jogos interaja com outros recursos da AWS .
- Permita que os processos do servidor de jogos obtenham informações sobre a frota em que estão executando.

Tópicos

- [Adicione Amazon GameLift Servers ao seu servidor de jogos](#)
- [Comunique-se com outros AWS recursos de suas frotas](#)

Adicione Amazon GameLift Servers ao seu servidor de jogos

Este tópico descreve como modificar o código do servidor de jogos para que os processos do servidor de jogos possam se comunicar com o Amazon GameLift Servers serviço. Use essas instruções para servidores de jogos que você planeja implantar em EC2 frotas Amazon GameLift Servers gerenciadas, frotas de contêineres gerenciados ou frotas em qualquer lugar.

Os processos do servidor do jogo se comunicam com o Amazon GameLift Servers serviço para receber instruções do serviço e relatar a integridade do processo do servidor e o status da sessão do jogo. Para obter informações detalhadas sobre as interações entre os componentes da solução de hospedagem de jogos (servidor de jogos, serviço de back-end, cliente de jogos e Amazon GameLift Servers), consulte [Interações cliente/servidor do jogo com Amazon GameLift Servers](#).

Para preparar seu jogo para hospedagem, adicione o SDK do servidor Amazon GameLift Servers ao seu projeto de servidor de jogos. Se você estiver usando o Amazon GameLift Servers plug-in para Unreal Engine ou Unity, o SDK do servidor está incorporado e pronto para uso. O Server SDK está disponível em várias linguagens. Para obter mais informações sobre o suporte de ferramentas para servidores de jogos, incluindo o SDK do servidor, consulte [Obtenha ferramentas Amazon GameLift Servers de desenvolvimento](#).

Referências da API do Server SDK:

- [SDK 5.x do servidor C++ para Amazon GameLift Servers -- Ações](#)
- [SDK 5.x do servidor C# para Amazon GameLift Servers -- Ações](#)
- [SDK 5.x do servidor C++ \(Unreal\) para -- Ações Amazon GameLift Servers](#)

- [SDK do servidor Go para Amazon GameLift Servers -- Ações](#)

Inicializar o processo do servidor

Adicione código para estabelecer comunicação com o Amazon GameLift Servers serviço e informe quando o processo do servidor do jogo estiver pronto para hospedar uma sessão de jogo. Esse código deve ser executado antes de qualquer Amazon GameLift Servers código.

1. Inicialize um cliente de Amazon GameLift Servers API `InitSdk()` chamando. [Se você estiver preparando seu servidor de jogos para ser executado em EC2 frotas Amazon GameLift Servers gerenciadas, use o padrão `InitSDK\(\)` \(C++\) \(C#\) \(Unreal\) \(Go\) \(C++\) sem parâmetros](#). O cliente da API gerencia a conexão com o Amazon GameLift Servers serviço para você.

 Se você estiver preparando seu servidor de jogos para uso em uma frota Amazon GameLift Servers Anywhere:

Inicialize o cliente Amazon GameLift Servers da API chamando `InitSdk()` com o seguinte: `ServerParameters`

- O URL do Websocket usado para se conectar ao servidor do jogo.
- O ID do processo usado para hospedar o servidor de jogos.
- O ID do computador que hospeda os processos do seu servidor de jogos.
- O ID da frota que contém sua computação Amazon GameLift Servers Anywhere.
- O token de autorização gerado pela Amazon GameLift Servers operação [GetComputeAuthToken](#).

2. Notifique o serviço de que o processo do servidor de jogos está pronto para hospedar uma sessão de jogo. Chame `ProcessReady()` ([C++](#)) ([C#](#)) ([Unreal](#)) ([Go](#)) (C++) com o seguinte. `ProcessParameters` Cada processo do servidor de jogo deve ser chamado `ProcessReady()` apenas uma vez.
 - O número da porta do processo do servidor. Quando o processo do servidor inicia uma sessão de jogo, ele fornece a porta para o Amazon GameLift Servers serviço, que atualiza as informações da sessão do jogo. Seu jogo pode recuperar essas informações e fornecê-las aos clientes do jogo, que as usam para se conectar ao processo do servidor e participar da sessão do jogo.

- Os locais dos arquivos que você Amazon GameLift Servers deseja armazenar para você. Isso pode incluir registros de sessão de jogo e outros arquivos que o processo do servidor gera durante uma sessão de jogo. Embora Amazon GameLift Servers salve esses arquivos temporariamente na computação em que o processo do servidor está sendo executado, eles só ficam disponíveis até que a instância seja encerrada. Você pode acessar os arquivos armazenados por meio do [Amazon GameLift Serversconsole](#) ou chamando a operação da Amazon GameLift Servers API [GetGameSessionLogUrl\(\)](#).

 Se você estiver preparando seu servidor de jogos para uso em uma frota de contêineres Amazon GameLift Servers gerenciados:

Você não precisa especificar parâmetros de registro para uma frota de contêineres. Em vez disso, envie a sessão do jogo e outros dados de registro para a saída padrão. As frotas de contêineres capturam automaticamente toda a saída padrão do contêiner como um fluxo de log.

- As seguintes funções de retorno de chamada permitem Amazon GameLift Servers enviar mensagens ou solicitações para um processo do servidor de jogos. Você deve implementar cada uma dessas funções no código do servidor do jogo. Para obter mais informações, consulte `ProcessParameters` ([C++](#)) ([C#](#)) ([Unreal](#)) ([Go](#)) .
- (Opcional) `onHealthCheck` — Amazon GameLift Servers chama essa função regularmente para solicitar um relatório de status de saúde do servidor.
- `onStartGameSession`— Amazon GameLift Servers chama essa função em resposta à solicitação do cliente [CreateGameSession\(\)](#).
- `onProcessTerminate`— Amazon GameLift Servers força o processo do servidor a parar, permitindo que ele seja desligado normalmente.
- (Opcional) `onUpdateGameSession` — Amazon GameLift Servers entrega um objeto de sessão de jogo atualizado ao servidor do jogo ou fornece uma atualização de status em uma solicitação de preenchimento de partida. O recurso [FlexMatchde preenchimento](#) requer esse retorno de chamada.

Você também pode configurar um servidor de jogos para que ele possa acessar com segurança outros AWS recursos que você possui ou controla. Para obter mais informações, consulte [Comuniquese com outros AWS recursos de suas frotas](#).

(Opcional) Reportar a integridade do processo do servidor

Adicione código ao seu servidor de jogo para implementar a função `onHealthCheck()` de retorno de chamada. Amazon GameLift Servers invoca esse método de retorno de chamada periodicamente para coletar métricas de saúde. Para implementar essa função de retorno de chamada, faça o seguinte:

- Avalie o status da integridade do processo do servidor. Por exemplo, convém informar o processo do servidor como não íntegro caso alguma dependência externa falhar.
- Conclua a avaliação de integridade e responda ao retorno de chamada dentro de 60 segundos. Se Amazon GameLift Servers não receber uma resposta nesse período, ele automaticamente considera que o processo do servidor não está íntegro.
- Retornar um valor Booleano: verdadeiro para íntegro, falso para não íntegro.

Se você não implementar um retorno de chamada de verificação de integridade, Amazon GameLift Servers considere que o processo do servidor está íntegro, a menos que o servidor não responda.

O Amazon GameLift Servers serviço usa a integridade do processo do servidor para encerrar processos não íntegros e liberar recursos. Se um processo do servidor continuar sendo reportado como não íntegro ou não responder por três verificações de integridade consecutivas, o serviço poderá encerrar o processo e iniciar um novo. O serviço coleta métricas sobre a integridade do processo do servidor de uma frota.

(Opcional) Obtenha um certificado TLS

e o processo do servidor estiver em execução em uma frota com a geração de certificado TLS ativada, você poderá recuperar o certificado para estabelecer uma conexão segura com um cliente de jogo e criptografar a comunicação entre cliente e servidor. Uma cópia do certificado é armazenada na instância. Para obter a localização do arquivo, chame `GetComputeCertificate()` ([C++](#)) ([C#](#)) ([Unreal](#)) ([Go](#)) .

Iniciar uma sessão de jogo

Adicione código para implementar a função `onStartGameSession` de retorno de chamada. Amazon GameLift Servers invoca esse retorno de chamada para iniciar uma sessão de jogo no processo do servidor.

A `onStartGameSession` função usa um [GameSession](#) objeto como parâmetro de entrada. Esse objeto inclui informações importantes sobre a sessão do jogo, como o número máximo de jogadores.

Também pode incluir dados do jogo e dados do jogador. A implementação da função deve realizar as seguintes tarefas:

- Encerrar ações para criar uma sessão de jogo com base nas propriedades `GameSession`. No mínimo, o servidor do jogo deve associar o ID da sessão do jogo, que os clientes do jogo referenciam ao se conectar ao processo do servidor.
- Processe os dados do jogo e os dados do jogador conforme necessário. Esses dados estão no objeto de `GameSession`.
- Notifique o Amazon GameLift Servers serviço quando uma nova sessão de jogo estiver pronta para aceitar jogadores. Chame a operação `ActivateGameSession()` da API do servidor ([C++](#)) ([C#](#)) ([Unreal](#)) ([Go](#)) . Em resposta a uma chamada bem-sucedida, o serviço altera o status da sessão do jogo para `ACTIVE`.

(Opcional) Validar um jogador

Se você estiver monitorando o status das sessões dos jogadores, adicione um código para validar um novo jogador quando ele se conectar a um servidor de jogo. Amazon GameLift Servers rastreia os jogadores atuais e os slots de sessão de jogo disponíveis.

Para validação, um cliente de jogo que tenta entrar em uma sessão de jogo deve incluir um ID de sessão de jogador. Amazon GameLift Servers gera esse ID quando seu jogo inicia novas sessões de jogo chamando [StartGameSessionPlacement\(\)](#) ou [StartMatchmaking\(\)](#). Nessas solicitações, um espaço aberto em uma sessão de jogo é reservado para a sessão do jogador.

Quando o processo do servidor do jogo recebe uma solicitação de conexão do cliente do jogo, ele chama `AcceptPlayerSession()` ([C++](#)) ([C#](#)) ([Unreal](#)) ([Go](#)) com o ID da sessão do jogador. Em resposta, Amazon GameLift Servers verifica se o ID da sessão do jogador corresponde a um espaço aberto reservado na sessão do jogo. Depois de Amazon GameLift Servers validar o ID da sessão do jogador, o processo do servidor aceita a conexão. O jogador pode então entrar na sessão do jogo. Se Amazon GameLift Servers não validar o ID da sessão do jogador, o processo do servidor nega a conexão.

(Opcional) Informar o término de uma sessão de jogador

Se você estiver acompanhando o status das sessões dos jogadores, adicione um código para notificar Amazon GameLift Servers quando um jogador sair da sessão do jogo. Esse código deve ser executado sempre que o processo do servidor detectar uma queda na conexão. Amazon GameLift

Server usa essa notificação para rastrear os jogadores atuais e os slots disponíveis na sessão do jogo.

Para lidar com conexões perdidas em seu código, adicione uma chamada à operação de API do servidor `RemovePlayerSession()` ([C++](#)) ([C#](#)) ([Unreal](#)) ([Go](#)) ([C++](#)) com o ID de sessão do jogador correspondente.

Encerrar uma sessão de jogo

Adicione código à sequência de desligamento do processo do servidor para notificar Amazon GameLift Servers quando uma sessão de jogo estiver terminando. Para reciclar e atualizar os recursos de hospedagem, encerre cada processo do servidor após a conclusão da sessão do jogo.

No início do código de desligamento do processo do servidor, chame a operação `ProcessEnding()` da API do servidor ([C++](#)) ([C#](#)) ([Unreal](#)) ([Go](#)) g. Essa chamada notifica Amazon GameLift Servers que o processo do servidor está sendo encerrado. Amazon GameLift Servers altera o status da sessão do jogo e o status do processo do servidor para `TERMINATED`. Após chamar `ProcessEnding()`, é seguro que o processo seja encerrado.

Responder a uma notificação de desligamento do processo do servidor

Adicione código para encerrar o processo do servidor em resposta a uma notificação do Amazon GameLift Servers serviço. O serviço envia essa notificação quando o processo do servidor relata consistentemente que não está íntegro ou se a instância em que o processo do servidor está sendo executado está sendo encerrada. Amazon GameLift Servers pode interromper uma instância como parte de um evento de redução de capacidade ou em resposta à interrupção da Instância Spot. A interrupção da Instância Spot fornece um aviso de dois minutos, o que dá tempo ao processo do servidor para desconectar os jogadores normalmente, preservar os dados do estado do jogo e realizar outras tarefas de limpeza.

Para lidar com uma notificação de desligamento, faça as seguintes alterações no código do servidor do jogo:

- [Implemente a função de retorno de chamada `onProcessTerminate\(\)`](#) ([C++](#)) ([C#](#)) ([Unreal](#)) ([Go](#)) ([C++](#)). Essa função deve chamar o código que encerra o processo do servidor.
- Chame a operação `GetTerminationTime()` da API do servidor ([C++](#)) ([C#](#)) ([Unreal](#)) ([Go](#)) a partir do código de desligamento do servidor do jogo. Se Amazon GameLift Servers tiver emitido uma chamada para interromper o processo do servidor, `GetTerminationTime()` retornará o tempo estimado de encerramento.

- No início do código de desligamento do servidor do jogo, chame a operação de API do servidor `ProcessEnding()` ([C++](#)) ([C#](#)) ([Unreal](#)) ([Go](#)) . Essa chamada notifica o serviço do Amazon GameLift Servers sobre o encerramento do processo do servidor. Em seguida, o serviço altera o status do processo do servidor para `TERMINATED`. Após chamar `ProcessEnding()`, é seguro que o processo seja encerrado.

Comunique-se com outros AWS recursos de suas frotas

Ao criar uma versão de servidor de jogos para implantação em Amazon GameLift Servers frotas, talvez você queira que os aplicativos em sua versão de jogo se comuniquem de forma direta e segura com outros AWS recursos que você possui. Como Amazon GameLift Servers gerencia suas frotas de hospedagem de jogos, você deve dar acesso Amazon GameLift Servers limitado a esses recursos e serviços.

Alguns exemplos de cenários incluem:

- Use um CloudWatch agente da Amazon para coletar métricas, registros e rastreamentos de EC2 frotas gerenciadas e frotas do Anywhere.
- Envie dados de log da instância para o Amazon CloudWatch Logs.
- Obtenha arquivos de jogos armazenados em um bucket do Amazon Simple Storage Service (Amazon S3).
- Ler e gravar dados de jogos (como modos de jogo ou inventário) armazenados em um banco de dados do Amazon DynamoDB ou em outro serviço de armazenamento de dados.
- Envie sinais diretamente para uma instância usando o Amazon Simple Queue Service (Amazon SQS).
- Acesse recursos personalizados que são implantados e executados no Amazon Elastic Compute Cloud (Amazon EC2).

Amazon GameLift Servers suporta esses métodos para estabelecer acesso:

- [Acesse AWS recursos com uma função do IAM](#)
- [Acesse AWS recursos com o peering de VPC](#)

Acesse AWS recursos com uma função do IAM

Use uma perfil do IAM para especificar quem pode acessar seus recursos e definir limites para esse acesso. Partes confiáveis podem “assumir” uma função e obter credenciais de segurança temporárias que as autorizem a interagir com os recursos. Quando as partes fazem solicitações de API relacionadas ao recurso, elas devem incluir as credenciais.

Para configurar o acesso controlado por um perfil do IAM, execute as seguintes tarefas:

1. [Crie o perfil do IAM](#)
2. [Modifique os aplicativos para adquirir credenciais](#)
3. [Associe uma frota ao perfil do IAM](#)

Crie o perfil do IAM

Nesta etapa, você cria uma função do IAM, com um conjunto de permissões para controlar o acesso aos seus AWS recursos e uma política de confiança que concede Amazon GameLift Servers direitos de uso das permissões da função.

Para obter instruções sobre como configurar o perfil do IAM, consulte [Configure uma função de serviço do IAM para Amazon GameLift Servers](#). Ao criar a política de permissões, escolha serviços, recursos e ações específicos com os quais seus aplicativos precisam trabalhar. Como melhor prática, limite o escopo das permissões o máximo possível.

Depois de criar a função, anote o nome do recurso da Amazon (ARN) da função. Você precisa da função do ARN durante a criação da frota.

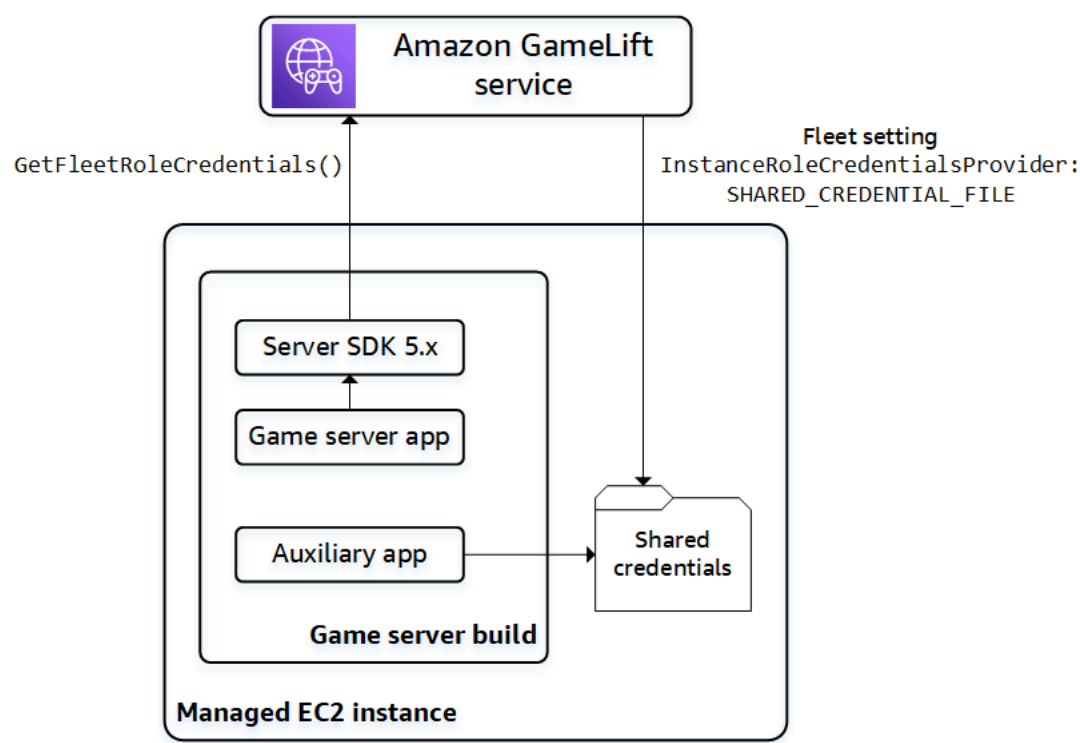
Modifique os aplicativos para adquirir credenciais

Nesta etapa, você configura seus aplicativos para adquirir credenciais de segurança para a função do IAM e usá-las ao interagir com seus AWS recursos. Consulte a tabela a seguir para determinar como modificar seus aplicativos com base (1) no tipo de aplicativo e (2) na versão do SDK do servidor com Amazon GameLift Servers a qual seu jogo usa para se comunicar.

	Aplicativos de servidor de jogos	Outros aplicativos
Usando o SDK do servidor versão 5.x	Chame o método SDK do servidor <code>GetFleetRoleCredentials()</code>	Adicione código ao aplicativo para extrair credenciais de um arquivo compartilhado na instância da frota.

	Aplicativos de servidor de jogos	Outros aplicativos
	a partir do código do servidor do jogo.	
Usando o SDK do servidor versão 4 ou anterior	Ligue para AWS Security Token Service (AWS STS) AssumeRole com a função ARN.	Ligue para AWS Security Token Service (AWS STS) AssumeRole com a função ARN.

Para jogos integrados ao SDK 5.x do servidor, esse diagrama ilustra como os aplicativos em sua versão de jogo implantada podem adquirir credenciais para o perfil do IAM.



Chamada **GetFleetRoleCredentials()** (SDK 5.x do servidor)

No código do servidor do jogo, que já deve estar integrado ao SDK 5.x do Amazon GameLift Servers servidor, chame `GetFleetRoleCredentials` ([C++](#)) ([C#](#)) ([Unreal](#)) ([Go](#)) para recuperar um conjunto de credenciais temporárias. Quando as credenciais expirarem, você poderá atualizá-las com outra chamada para `GetFleetRoleCredentials`.

Use credenciais compartilhadas (SDK 5.x do servidor)

Para aplicativos que não são de servidor que são implantados com compilações de servidores de jogos usando o SDK 5.x do servidor, adicione código para obter e usar credenciais armazenadas em um arquivo compartilhado. Amazon GameLift Servers gera um perfil de credenciais para cada instância da frota. As credenciais estão disponíveis para uso por todos os aplicativos na instância. Amazon GameLift Servers atualiza continuamente as credenciais temporárias.

Você deve configurar uma frota para gerar o arquivo de credenciais compartilhadas na criação da frota.

Em cada aplicativo que precisa usar o arquivo de credenciais compartilhado, especifique a localização do arquivo e o nome do perfil, da seguinte forma:

Windows:

```
[credentials]
shared_credential_profile= "FleetRoleCredentials"
shared_credential_file= "C:\\Credentials\\credentials"
```

Linux

```
[credentials]
shared_credential_profile= "FleetRoleCredentials"
shared_credential_file= "/local/credentials/credentials"
```

Exemplo: configurar um CloudWatch agente para coletar métricas para instâncias de Amazon GameLift Servers frota

Se você quiser usar um CloudWatch agente da Amazon para coletar métricas, registros e rastreamentos de suas Amazon GameLift Servers frotas, use esse método para autorizar o agente a emitir os dados para sua conta. Nesse cenário, execute as seguintes etapas:

1. Recupere ou grave o `config.json` arquivo do CloudWatch agente.
2. Atualize o arquivo `common-config.toml` para que o atendente identifique o nome do arquivo de credenciais e o nome do perfil, conforme descrito acima.
3. Configure seu script de instalação de compilação do servidor de jogos para instalar e iniciar o CloudWatch agente.

Uso **AssumeRole()** (SDK do servidor 4)

Adicione código aos seus aplicativos para assumir a função do IAM e obter credenciais para interagir com seus AWS recursos. Qualquer aplicativo executado em uma instância de Amazon GameLift Servers frota com o SDK 4 do servidor ou anterior pode assumir a função do IAM.

No código do aplicativo, antes de acessar um AWS recurso, o aplicativo deve chamar a operação da [AssumeRole](#) API AWS Security Token Service (AWS STS) e especificar o ARN da função. Essa operação retorna um conjunto de credenciais temporárias que autoriza o aplicativo a acessar o AWS recurso. Para obter mais informações, consulte [Uso de credenciais temporárias com AWS recursos](#) no Guia do usuário do IAM.

Associe uma frota ao perfil do IAM

Depois de criar o perfil do IAM e atualizar os aplicativos na versão do seu servidor de jogos para obter e usar as credenciais de acesso, você pode implantar uma frota. Quando você configura a nova frota, defina os seguintes parâmetros:

- [InstanceRoleArn](#)— Defina esse parâmetro como o ARN da função do IAM.
- [InstanceRoleCredentialsProvider](#)— Para solicitar Amazon GameLift Servers a geração de um arquivo de credenciais compartilhado para cada instância da frota, defina `SHARED_CREDENTIAL_FILE` esse parâmetro como.

Você deve definir esses valores ao criar a frota. Eles não podem ser atualizados mais tarde.

Acesse AWS recursos com o peering de VPC

Você pode usar o emparelhamento da Amazon Virtual Private Cloud (Amazon VPC) para se comunicar entre aplicativos em execução em uma Amazon GameLift Servers instância e outro recurso. AWS Uma VPC é uma rede privada virtual que você define e que inclui um conjunto de recursos gerenciados por meio do seu. Conta da AWS Cada frota do Amazon GameLift Servers tem sua própria VPC. Com o peering de VPC, você pode estabelecer uma conexão de rede direta entre a VPC para sua frota e para seus outros recursos. AWS

O Amazon GameLift Servers simplifica o processo de configuração de conexões de emparelhamento de VPCs para seus servidores de jogos. Ele lida com solicitações de emparelhamento, atualiza tabelas de rota e configura as conexões, conforme necessário. Para obter instruções sobre como configurar o emparelhamento da VPC para os servidores de jogos, consulte [Emparelhamento de VPC para Amazon GameLift Servers](#).

Integre seu cliente de jogo com Amazon GameLift Servers

Os tópicos desta seção descrevem o gerenciado Amazon GameLift Servers funcionalidade que você pode adicionar a um serviço de back-end. Um serviço de back-end executa as seguintes tarefas:

- Solicita informações sobre sessões de jogo ativas de Amazon GameLift Servers.
- Adiciona um jogador a uma sessão de jogo existente.
- Cria uma sessão de jogo e adiciona jogadores a ela.
- Altera metadados para uma sessão de jogo existente.

Para obter mais informações sobre como os clientes do jogo interagem com Amazon GameLift Servers e servidores de jogos em execução em Amazon GameLift Servers, consulte [Interações cliente/servidor do jogo com Amazon GameLift Servers](#).

Pré-requisitos

- Um Conta da AWS.
- Uma compilação de servidor de jogos enviada para Amazon GameLift Servers.
- Uma frota para hospedar os jogos.

Tópicos

- [Adicione Amazon GameLift Servers ao seu cliente de jogo](#)
- [Gerar jogador IDs](#)

Adicione Amazon GameLift Servers ao seu cliente de jogo

Amazon GameLift ServersIntegre-se aos componentes do jogo que precisam de informações sobre a sessão do jogo, crie novas sessões de jogo e adicione jogadores aos jogos. Dependendo da arquitetura do jogo, essa funcionalidade está serviços de back-end que processam tarefas como autenticação do jogador, marcação de jogos ou colocação da sessão de jogo.

Note

Para obter informações detalhadas sobre como configurar o matchmaking para seu jogo, consulte o [Guia do Amazon GameLift ServersFlexMatch desenvolvedor](#).

Configurar Amazon GameLift Servers em um serviço de back-end

Adicione código para inicializar um Amazon GameLift Servers cliente e armazenar as configurações da chave. Esse código deve ser executado antes de qualquer código dependente de Amazon GameLift Servers.

1. Defina uma configuração de cliente. Use a configuração padrão do cliente ou crie um objeto de configuração do cliente personalizado. Para obter mais informações, consulte [AWS::Client::ClientConfiguration](#)(C++) ou [AmazonGameLiftConfig](#)(C#).

A configuração do cliente especifica uma região de destino e um endpoint a serem usados ao entrar em contato. Amazon GameLift Servers A região identifica o conjunto de recursos implantados (frotas, filas e marcador de jogos) a serem usados. A configuração do cliente padrão define a localização na região Leste dos EUA (Norte da Virgínia). Para usar qualquer outra Região, crie uma configuração personalizada.

2. Inicialize um cliente do Amazon GameLift Servers. Use [Aws::GameLift::GameLiftClient\(\)](#) (C++) ou [AmazonGameLiftClient\(\)](#) (C#) com uma configuração de cliente padrão ou uma configuração de cliente personalizada.
3. Adicione um mecanismo para gerar um identificador exclusivo para cada jogador. Para obter mais informações, consulte [Gerar jogador IDs](#).
4. Colete e armazene informações a seguir:
 - Frota de destino — muitas solicitações de Amazon GameLift Servers API devem especificar uma frota. Para fazer isso, use um ID de frota ou alias que aponte para a frota de destino. Como melhor prática, use os aliases de frota para poder trocar jogadores de uma frota para outra sem precisar atualizar os serviços de back-end.
 - Fila de destino – Para jogos que usam filas de várias frotas para colocar novas sessões de jogo, especifique o nome da fila a ser usada.
 - AWS credenciais — Todas as chamadas para Amazon GameLift Servers devem fornecer credenciais para quem hospeda Conta da AWS o jogo. Você adquire essas credenciais criando um usuário jogador, conforme descrito em [Configurar o acesso programático para o jogo](#). Dependendo de como você gerencia o acesso do usuário player, faça o seguinte:
 - Se você usa uma função para gerenciar as permissões de usuário do jogador, adicione código para assumir a função antes de chamar uma Amazon GameLift Servers API. A solicitação para assumir a função retorna um conjunto de credenciais de segurança temporárias. Para obter mais informações, consulte [Mudar para uma função do IAM \(AWS API\)](#) no Guia do usuário do IAM.

- Caso tenha credenciais de segurança de longo prazo, configure seu código para localizar e use as credenciais armazenadas. Consulte [Autenticar usando credenciais de longo prazo](#) no Guia de referência de ferramentas AWS SDKs e ferramentas. Para obter informações sobre como armazenar credenciais, consulte as referências de AWS API para [\(C++\)](#) e [\(.NET\)](#).
- Se você tiver credenciais de segurança temporárias, adicione código para atualizar regularmente as credenciais usando o AWS Security Token Service (AWS STS), conforme descrito em [Usando credenciais de segurança temporárias com o AWS SDKs no Guia do usuário do IAM](#). O código deve solicitar novas credenciais antes que as antigas expirem.

Obter sessões de jogos

Adicione o código para descobrir as sessões de jogos disponíveis e gerenciar as configurações de sessões de jogos e metadados.

Procure sessões de jogos ativas

Use [SearchGameSessions](#) para obter informações sobre uma sessão de jogo específica, todas as sessões ativas ou sessões que atendem a um conjunto de critérios de pesquisa. Essa chamada retorna um [GameSession](#) objeto para cada sessão de jogo ativa que corresponde à sua solicitação de pesquisa.

Use critérios de pesquisa para obter uma lista filtrada de sessões de jogos ativas para jogadores ingressarem. Por exemplo, você pode filtrar sessões da seguinte maneira:

- Exclua sessões de jogos que sejam completas: `CurrentPlayerSessionCount = MaximumPlayerSessionCount`.
- Escolha sessões de jogos no tempo em que a sessão esteve em execução: avalie `CreationTime`.
- Encontre sessões de jogos baseadas em uma propriedade personalizada do jogo: `gameSessionProperties.gameMode = "brawl"`.

Gerencie sessões de jogos

Use qualquer uma das operações a seguir para recuperar ou atualizar as informações sobre a sessão do jogo.

- [DescribeGameSessionDetails\(\)](#) — Obtenha o status de proteção de uma sessão de jogo, além das informações da sessão de jogo.
- [UpdateGameSession\(\)](#) — Altere os metadados e as configurações de uma sessão de jogo conforme necessário.
- [GetGameSessionLogUrl](#) — Acesse os registros armazenados da sessão do jogo.

Criar sessões de jogos

Adicione um código para iniciar novas sessões de jogos nas frotas implantadas e disponibilizá-las para jogadores. Há duas opções para criar sessões de jogo, dependendo se você está implantando seu jogo em várias regiões da AWS ou em uma única região.

Crie uma sessão de jogo em uma fila com vários locais

Use [StartGameSessionPlacement](#) para fazer uma solicitação para uma nova sessão de jogo em uma fila. Para usar essa operação, crie uma fila. Isso determina Amazon GameLift Servers onde a nova sessão de jogo será colocada. Para obter informações mais sobre filas e como usá-las, consulte [Gerenciando o posicionamento da sessão de jogo com Amazon GameLift Servers filas](#).

Ao criar uma colocação de sessão do jogo, especifique o nome da fila a ser usada, um nome da sessão do jogo, um número máximo de jogadores simultâneos e um conjunto opcional de propriedades do jogo. Opcionalmente, também é possível fornecer uma lista de jogadores para ingressar automaticamente na sessão do jogo. Se você incluir dados de latência do jogador para regiões relevantes, Amazon GameLift Servers use essas informações para colocar a nova sessão de jogo em uma frota que ofereça a experiência de jogo ideal para os jogadores.

Para obter medições precisas de latência, use os Amazon GameLift Servers beacons de ping UDP. Esses endpoints permitem medir a latência real da rede UDP entre dispositivos de jogadores e possíveis locais de hospedagem, resultando em decisões de posicionamento mais precisas do que o uso de pings ICMP. Para obter mais informações sobre o uso de beacons de ping UDP para medir a latência, consulte [Sinalizadores de ping UDP](#).

A colocação da sessão do jogo é um processo assíncrono. Após ter feito uma solicitação, você poderá deixar que ela seja bem-sucedida ou expire. Você também pode cancelar a solicitação a qualquer momento usando [StopGameSessionPlacement](#). Para verificar o status da sua solicitação de colocação, ligue [DescribeGameSessionPlacement](#).

Crie uma sessão do jogo em uma frota específica

Use [CreateGameSession](#) para criar uma nova sessão em uma frota especificada. Essa operação síncrona pode ser bem-sucedida ou falhar, dependendo dos recursos que a frota tem à disposição para hospedar uma nova sessão do jogo. Depois de Amazon GameLift Servers criar a nova sessão de jogo e devolver um [GameSession](#) objeto, você pode juntar jogadores a ela.

Ao usar esta operação, forneça um ID de frota ou alias, um nome de sessão e um número máximo de jogadores simultâneos para o jogo. Você também pode incluir um conjunto de propriedades do jogo. As propriedades do jogo são definidas em uma matriz de pares de chave-valor.

Se você usar o Amazon GameLift Servers recurso de proteção de recursos para limitar o número de sessões de jogo que um jogador pode criar, forneça o ID de jogador do criador da sessão de jogo.

Adicionar um jogador a uma sessão de jogo

Adicione código para reservar um slot de jogador em uma sessão de jogo ativa e conecte clientes de jogos a sessões de jogos.

1. Reserve um slot de jogador em uma sessão de jogo

Para reservar um slot de jogador, crie uma nova sessão de jogador para a sessão de jogo. Para obter mais informações sobre sessões de jogadores, consulte [Como os jogadores se conectam aos jogos](#).

Há duas maneiras de criar novas sessões de jogador:

- Use [StartGameSessionPlacement](#) para reservar vagas para um ou mais jogadores na nova sessão de jogo.
- Reserve vagas de jogador para um ou mais jogadores usando [CreatePlayerSession](#) ou [CreatePlayerSessions](#) com um ID de sessão de jogo.

Amazon GameLift Servers primeiro verifica se a sessão de jogo está aceitando novos jogadores e tem vagas disponíveis para jogadores. Se for bem-sucedido, Amazon GameLift Servers reserva um espaço para o jogador, cria a nova sessão de jogador e retorna um [PlayerSession](#) objeto. Esse objeto contém o nome DNS, o endereço IP e a porta que um cliente de jogo precisa para se conectar à sessão do jogo.

Uma solicitação de sessão de jogador deve incluir um ID exclusivo para cada jogador. Para obter mais informações, consulte [Gerar jogador IDs](#).

Uma sessão de jogador pode incluir um conjunto de dados de jogador personalizado. Esses dados são armazenados no objeto de sessão do player recém-criado, que você pode recuperar chamando [DescribePlayerSessions\(\)](#). Amazon GameLift Server também passa esse objeto para o servidor do jogo quando o jogador se conecta diretamente à sessão do jogo. Ao solicitar várias sessões de jogadores, forneça uma sequência de dados do jogador para cada jogador mapeado para o ID do jogador na solicitação.

2. Conecte-se a uma sessão do jogo

Adicione código ao cliente do jogo para recuperar o objeto de `PlayerSession`, que contém as informações de conexão da sessão do jogo. Use essas informações para estabelecer uma conexão direta com o servidor.

- Conecte-se usando a porta especificada e o nome DNS ou endereço IP atribuído ao processo do servidor.
- Se a geração de certificados TLS estiver habilitada para suas frotas, deverá se conectar usando o nome DNS e a porta.
- Se o seu servidor de jogo valida as conexões de entrada dos jogadores, faça referência ao ID da sessão do jogador.

Depois de fazer a conexão, o processo do cliente e do servidor do jogo se comunicam diretamente sem envolvimento Amazon GameLift Servers. O servidor mantém a comunicação Amazon GameLift Servers para relatar o status da conexão do jogador, o status de saúde e muito mais. Se o servidor do jogo valida os novos jogadores, ele verifica se o ID da sessão do jogador corresponde a um espaço reservado na sessão do jogo e aceita ou nega a conexão do jogador. Quando o jogador se desconectar, o processo do servidor informará a conexão desfeita.

Use as propriedades da sessão do jogo

O cliente de jogo pode passar dados para uma sessão de jogo usando uma propriedade do jogo. As propriedades do jogo são pares de valores-chave que o servidor de jogo pode adicionar, ler, listar e alterar. Você pode transmitir uma propriedade do jogo ao criar uma sessão de jogo ou posteriormente quando a sessão do jogo estiver ativa. Uma sessão de jogo pode conter até 16 propriedades do jogo. Não é possível excluir as propriedades do jogo.

Por exemplo, o jogo oferece os seguintes níveis de dificuldade: Novice, Easy, Intermediate e Expert. Um jogador escolhe Easy e depois começa o jogo. Seu cliente de jogo solicita uma nova sessão de jogo Amazon GameLift Servers usando uma `StartGameSessionPlacement` ou `CreateGameSession` conforme explicado nas seções anteriores. Na solicitação, o cliente informa o seguinte: `{"Key": "Difficulty", "Value": "Easy"}`.

Em resposta à solicitação, Amazon GameLift Servers cria um `GameSession` objeto que contém a propriedade do jogo especificada. Amazon GameLift Servers em seguida, instrui um servidor de jogo disponível a iniciar a nova sessão de jogo e passar o `GameSession` objeto. O servidor do jogo inicia uma sessão de jogo com `Difficulty` definida como Easy.

Saiba mais

- [GameProperty tipo de dados](#)
- [SearchGameSessions\(\) exemplos](#)
- [UpdateGameSession\(\) GameProperties parâmetro](#)

Gerar jogador IDs

Amazon GameLift Servers usa uma sessão de jogador para representar um jogador conectado a uma sessão de jogo. Amazon GameLift Servers cria uma sessão de jogador sempre que um jogador se conecta a uma sessão de jogo usando um cliente de jogo integrado com Amazon GameLift Servers. Quando um jogador sai de um jogo, a sessão do jogador termina. Amazon GameLift Servers não reutiliza sessões de jogadores.

O exemplo de código a seguir gera aleatoriamente um jogador IDs único:

```
bool includeBrackets = false;
bool includeDashes = true;
string playerId = AZ::Uuid::CreateRandom().ToString<string>(includeBrackets,
    includeDashes);
```

Para obter mais informações sobre sessões de jogadores, consulte [Sessões de jogos e jogadores no Amazon GameLift Servers console](#).

Motores de jogo e Amazon GameLift Servers

Você pode usar o gerenciado Amazon GameLift Servers serviço com a maioria dos principais mecanismos de jogos que suportam bibliotecas C++ ou C#, incluindo O3DE, Unreal Engine e

Unity. Compile a versão que você precisa para o seu jogo; consulte os arquivos LEIA-ME de cada versão para mais instruções de compilação e requisitos mínimos. Para obter mais informações sobre os disponíveis Amazon GameLift Servers SDKs, plataformas de desenvolvimento e sistemas operacionais compatíveis, consulte [Obtenha ferramentas Amazon GameLift Servers de desenvolvimento](#) os servidores de jogos.

Além das informações específicas do motor fornecidas neste tópico, encontre ajuda adicional com a integração Amazon GameLift Servers em seus servidores de jogos, clientes e serviços nos seguintes tópicos:

- [Adicione Amazon GameLift Servers ao seu servidor de jogos](#)— Instruções detalhadas sobre a integração Amazon GameLift Servers em um servidor de jogos.
- [Adicione Amazon GameLift Servers ao seu cliente de jogo](#) – Instruções detalhadas sobre a integração a um cliente ou serviço de jogo, inclusive a criação de sessões de jogos e a entrada de jogadores em jogos.

O3DE

Servidores de jogos

Prepare seus servidores de jogos para hospedagem em Amazon GameLift Servers usando o SDK [do servidor para Amazon GameLift Servers para C++](#). Consulte [Adicione Amazon GameLift Servers ao seu servidor de jogos](#) para obter ajuda na integração da funcionalidade necessária ao servidor de jogos.

Clientes e serviços de jogos

Permita que seus clientes e/ou serviços de jogos interajam com Amazon GameLift Servers serviço, como encontrar sessões de jogo disponíveis ou criar novas e adicionar jogadores aos jogos. A principal funcionalidade do cliente é fornecida no [AWS SDK for C++](#). Para integrar Amazon GameLift Servers em seu projeto de jogo O3DE, veja e. [Adicionar Amazon GameLift Servers para um cliente e servidor de jogos O3DE](#) [Adicione Amazon GameLift Servers ao seu cliente de jogo](#)

Unreal Engine

Servidores de jogos

Prepare seus servidores de jogos para hospedagem em Amazon GameLift Servers adicionando o SDK [do servidor para Amazon GameLift Servers para o Unreal Engine](#) em seu projeto e implementando a funcionalidade de servidor necessária. Para obter ajuda na configuração do plug-in

Unreal Engine e na adição Amazon GameLift Servers código, veja [Amazon GameLift ServersIntegre-se a um projeto Unreal Engine](#).

Clientes e serviços de jogos

Permita que seus clientes e/ou serviços de jogos interajam com Amazon GameLift Servers serviço, como encontrar sessões de jogo disponíveis ou criar novas e adicionar jogadores aos jogos. A principal funcionalidade do cliente é fornecida no [AWS SDK for C++](#). Para integrar Amazon GameLift Servers em seu projeto de jogo Unreal Engine, consulte [Adicione Amazon GameLift Servers ao seu cliente de jogo](#).

Unity

Servidores de jogos

Prepare seus servidores de jogos para hospedagem em Amazon GameLift Servers adicionando o SDK [do servidor para Amazon GameLift Servers para C#](#) em seu projeto e implementando a funcionalidade de servidor necessária. Para obter ajuda na configuração do Unity e na adição Amazon GameLift Servers código, veja [Amazon GameLift ServersIntegre-se a um projeto Unity](#).

Clientes e serviços de jogos

Permita que seus clientes e/ou serviços de jogos interajam com Amazon GameLift Servers serviço, como encontrar sessões de jogo disponíveis ou criar novas e adicionar jogadores aos jogos. A principal funcionalidade do cliente é fornecida no [AWS SDK para .NET](#). Para integrar Amazon GameLift Servers em seu projeto de jogo Unity, consulte [Adicione Amazon GameLift Servers ao seu cliente de jogo](#).

Outros mecanismos

Para obter uma lista completa dos Amazon GameLift Servers SDKs disponível para servidores e clientes de jogos, consulte [the section called “Obter ferramentas de desenvolvimento”](#).

Adicionar Amazon GameLift Servers para um cliente e servidor de jogos O3DE

Você pode usar o O3DE, um mecanismo 3D de código aberto, multiplataforma e em tempo real para criar experiências interativas de alto desempenho, incluindo jogos e simulações. O renderizador e as ferramentas O3DE são agrupados em uma estrutura modular que você pode modificar e estender com suas ferramentas de desenvolvimento preferidas.

A estrutura modular usa Gems que contêm bibliotecas com interfaces e ativos padrão. Selecione seus próprios Gems para escolher quais funcionalidades adicionar com base em seus requisitos.

A ferramenta Amazon GameLift Servers O Gem fornece os seguintes recursos:

Amazon GameLift Servers integração

Uma estrutura para estender a camada de rede O3DE e permitir que o Multiplayer Gem funcione com o Amazon GameLift Servers solução de servidor dedicado. O Gem fornece integrações com o SDK do [servidor para Amazon GameLift Server](#) e o cliente AWS SDK (para chamar o Amazon GameLift Servers serviço em si).

Gerenciamento de compilação e pacotes

Instruções para empacotar e, opcionalmente, carregar a versão do servidor dedicado e um aplicativo AWS Cloud Development Kit (AWS CDK) (AWS CDK) para configurar e atualizar recursos.

Amazon GameLift Servers Configuração de gemas

Siga os procedimentos nesta seção para configurar o Amazon GameLift Servers Gem em O3DE.

Pré-requisitos

- Configure sua AWS conta para Amazon GameLift Servers. Para obter mais informações, consulte [Configurar um Conta da AWS](#).
- Configure as AWS credenciais para o O3DE. Para obter mais informações, consulte [Configurando AWS credenciais](#).
- Configure o AWS CLI AWS CDK e. Para obter mais informações, [AWS Command Line Interface](#) e [AWS Cloud Development Kit \(AWS CDK\)](#).

Ligue o Amazon GameLift Servers Gem e suas dependências

1. Abra o Gerente de projetos.
2. Abra o menu abaixo do seu projeto e escolha Editar configuração do projeto....
3. Selecione a opção Configurar o Gems.
4. Ligue o Amazon GameLift Servers Gem e as seguintes gemas dependentes:
 - [AWS Core Gem](#) — Forneça a estrutura a ser usada Serviços da AWS no O3DE.
 - [Multiplayer Gem](#) – Fornece funcionalidade multijogador ao estender a estrutura de rede.

Inclua o Amazon GameLift Servers Biblioteca estática de gemas

1. Inclua o `Gem::AWSGameLift.Server.Static` como `BUILD_DEPENDENCIES` para o destino do seu servidor de projeto.

```
ly_add_target(
    NAME YourProject.Server.Static STATIC
    ...
    BUILD_DEPENDENCIES
    PUBLIC
    ...
    PRIVATE
    ...
    Gem::AWSGameLift.Server.Static
)
```

2. `AWSGameLiftServiceDefina` como necessário para o componente do sistema do servidor de projetos.

```
void
YourProjectServerSystemComponent::GetRequiredServices(AZ::ComponentDescriptor::DependencyA
required)
{
    ...
    required.push_back(AZ_CRC_CE("AWSGameLiftServerService"));
    ...
}
```

3. (Opcional) Para fazer Amazon GameLift Servers solicitações de serviço em C++, inclua `Gem::AWSGameLift.Client.Static` na meta `BUILD_DEPENDENCIES` para seu cliente.

```
ly_add_target(
    NAME YourProject.Client.Static STATIC
    ...
    BUILD_DEPENDENCIES
    PUBLIC
    ...
    PRIVATE
    ...
    Gem::AWSCore.Static
    Gem::AWSGameLift.Client.Static
}
```

Integre seu jogo e servidor dedicado

Gerencie sessões de jogo em seu jogo e em um servidor de jogo dedicado com a [Integração de gerenciamento de sessões](#). Para apoiar FlexMatch, consulte [FlexMatch Integração](#).

Amazon GameLift ServersIntegre-se a um projeto Unreal Engine

Aprenda a integrar o Amazon GameLift Servers SDK do Unreal Engine em seus projetos de jogos para acessar o conjunto completo de recursos do SDK do servidor.

Dica

Para uma implantação mais rápida, experimente o plug-in Amazon GameLift Servers independente do Unreal Engine. Ele fornece fluxos de trabalho de interface de usuário guiados para implantar rapidamente seu servidor de jogos com configuração mínima, para que você possa testar os componentes do jogo em ação. Consulte [Amazon GameLift Serversplugin para Unreal Engine](#).

Recursos adicionais:

- [SDK 5.x do servidor C++ \(Unreal\) para -- Ações Amazon GameLift Servers](#)
- [Obtenha ferramentas Amazon GameLift Servers de desenvolvimento](#)

Instale o SDK do servidor para Unreal

Obtenha o Amazon GameLift Servers SDK de código aberto para Unreal Engine em [GitHub](#). Os arquivos readme do repositório contêm pré-requisitos e instruções de instalação.

Configure metas de construção e regras de módulo

Modifique seus arquivos de projeto de jogo para gerar corretamente os componentes de compilação para uso comAmazon GameLift Servers.

Para adicionar destinos de compilação de cliente e servidor:

1. Abra os arquivos de código do seu projeto de jogo e localize o `.../Games/[your application name]/Source/[your application name]/Target.cs` arquivo. Exemplo: `.../Source/GameLiftUnrealAppTarget.cs`. (Se você usa o Visual Studio, abra o `.sln` arquivo do projeto.)

2. Copie esse arquivo para criar dois novos arquivos de destino no Source/ diretório.

- Destino do cliente — Renomeie o novo arquivo para *[your application name]*Client.Target.cs. Edite o conteúdo para atualizar os valores do nome da classe e do tipo de destino, conforme ilustrado no código de exemplo a seguir:

```
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");
    }
}
```

- Destino do servidor — Renomeie o novo arquivo para *[your application name]*Server.Target.cs. Edite o conteúdo para atualizar os valores do nome da classe e do tipo de destino, conforme ilustrado no código de exemplo a seguir:

```
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. Regenere seus arquivos de projeto. Se você estiver usando o Visual Studio, você pode clicar com o botão direito do mouse no `.uproject` arquivo do seu projeto de jogo e selecionar Gerar arquivos de projeto do Visual Studio.

Para atualizar as regras do módulo do projeto do jogo:

Atualize as regras do módulo do projeto do jogo para depender do plug-in.

1. Abra os arquivos de código do seu projeto de jogo e localize o `.../Games/[your application name]/Source/[your application name].Build.cs` arquivo. Exemplo: `.../Source/GameLiftUnrealApp.Build.cs`. (Se você usa o Visual Studio, abra o `.sln` arquivo do projeto.)
2. Localize a `ModuleRules` classe e atualize conforme ilustrado no código de exemplo a seguir:

```
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",
"GameLiftServerSDK" });

        bEnableExceptions = true;
    }
}
```

3. Depois de criar os novos arquivos de destino e modificar as regras do módulo, reconstrua seu projeto de jogo.

Adicione a funcionalidade de hospedagem de jogos ao código do seu servidor

Após a instalação e configuração do SDK do servidor, a próxima etapa é integrar a funcionalidade de hospedagem de jogos ao código do servidor. O SDK do servidor permite que seu servidor de jogos se comunique com o Amazon GameLift Servers serviço, receba instruções para sessões de jogo, relate status e integridade e execute outras ações.

Este tópico fornece um exemplo de código que adiciona a funcionalidade mínima necessária para hospedar seu jogoAmazon GameLift Servers.

Etapa 1: atualizar o arquivo de **GameMode** cabeçalho

1. Abra os arquivos de código do seu projeto de jogo e localize o *Your-application-name*GameMode.h arquivo. Exemplo: GameLiftUnrealAppGameMode.h. Se você usa o Visual Studio, abra o .sln arquivo do seu projeto de jogo.
2. Altere o arquivo de cabeçalho para incluir o código de exemplo a seguir. Certifique-se de substituir "GameLiftUnrealApp" pelo nome do seu próprio aplicativo.

Código de exemplo para 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;  
};
```

Etapa 2: adicione as chamadas necessárias do SDK do servidor ao código do servidor do jogo

Use o código de amostra nesta seção para integrar o código do servidor do jogo para uso com Amazon GameLift Servers. Para obter detalhes sobre o que o código faz, consulte [Inicializar o processo do servidor SDK 5.x do servidor C++ \(Unreal\) para -- Ações Amazon GameLift Servers](#) e.

 Note

O WITH_GAMELIFT sinalizador de pré-processador tem duas finalidades:

- Restringe as chamadas de API de Amazon GameLift Servers back-end somente para compilações de servidores Unreal
- Garante a compatibilidade entre diferentes destinos de compilação do Unreal

1. Abra o arquivo *Your-application-name*GameMode.cpp de origem relacionado. Em nosso exemplo:GameLiftUnrealAppGameMode.cpp.
2. Altere o código para alinhá-lo com o código de exemplo a seguir. Certifique-se de substituir qualquer instância de "GameLiftUnrealApp" pelo nome do seu próprio aplicativo.

O exemplo de código fornecido mostra como adicionar os elementos necessários para integração com Amazon GameLift Servers o. Isso inclui:

- Inicialize um cliente de Amazon GameLift Servers API.
- Implemente funções de retorno de chamada para responder às solicitações do Amazon GameLift Servers serviçoOnStartGameSession, incluindoOnProcessTerminate, e. onHealthCheck
- Ligue ProcessReady() para notificar o Amazon GameLift Servers serviço quando estiver pronto para hospedar sessões de jogo.

Código de exemplo para 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();

#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"));
    GameLiftSdkModule->ProcessEnding();
}));

//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("GameServerLog/Saved/Logs/GameServerLog.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
}
```

Etapa 3: reconstruir o projeto do jogo

- Crie um projeto de jogo para os dois tipos de alvo a seguir: Editor de Desenvolvimento e Servidor de Desenvolvimento.

Note

Não é necessário reconstruir a solução. Em vez disso, crie o projeto na /Games/ pasta do seu aplicativo. Caso contrário, o Visual Studio reconstruirá todo o UE5 projeto, o que pode levar até uma hora.

Package seu servidor de jogos para hospedagem

Com o código do servidor do jogo agora integrado à funcionalidade mínima necessária do SDK do servidor, você está pronto para empacotar a versão do seu servidor de jogos usando o Unreal Editor.

Para empacotar a compilação do servidor do jogo

1. Abra o projeto do jogo no Unreal Editor.

2. Siga as etapas do Unreal Editor para empacotar seu servidor de jogo:

- Escolha sua plataforma de destino (Windows ou Linux).
- Selecione o destino de compilação do seu servidor (*[your application name]*Server).

O processo de empacotamento gera o executável do seu servidor de jogos: *[your application name]*Server.exe.

3. Prepare a versão do seu servidor de jogos para implantação em recursos de hospedagem. A compilação deve incluir os seguintes arquivos:

- Seu executável do servidor de jogos
- Para compilações do Windows, inclua esses arquivos (você pode encontrá-los na versão original do 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\)
- Todas as outras dependências necessárias para seu servidor de jogos.
- Bibliotecas OpenSSL, se necessário. Verifique a versão do SDK do servidor que está integrada à versão do seu servidor de jogos. Com o SDK do servidor para Unreal, versão 5.3.x, você não precisa mais incluir manualmente as bibliotecas do OpenSSL na versão do seu servidor de jogos. Para todas as versões anteriores, você deve localizar e adicionar os arquivos da biblioteca. [Veja a versão mais recente do SDK do servidor.](#)

Bibliotecas OpenSSL necessárias para servidores de jogos criados com o SDK de servidor para Unreal, versões 5.2.x e anteriores

Você deve incluir a mesma versão das bibliotecas do OpenSSL que foram usadas ao empacotar o servidor do jogo no Unreal. Essas bibliotecas estão localizadas na fonte do seu mecanismo de jogo. A localização varia de acordo com seu ambiente de desenvolvimento:

No 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

No 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

Copie as bibliotecas do OpenSSL para o pacote de compilação do jogo no mesmo diretório do arquivo executável do servidor do jogo.

Próximas etapas

Agora que você preparou uma versão de servidor de jogos com a funcionalidade mínima necessária para hospedagem Amazon GameLift Servers, considere estas possíveis próximas etapas:

- Implante seu servidor de jogos integrado para testes e desenvolvimento. Com uma frota Anywhere, você pode configurar sua máquina local como um recurso de hospedagem e usá-la para testar as conexões do servidor e do cliente do jogo. Para hospedagem baseada em nuvem, implante seu servidor de jogos em uma frota de contêineres gerenciada EC2 ou gerenciada. Consulte estes tópicos para obter orientação:
 - [Configure para desenvolvimento iterativo com Amazon GameLift Servers Anywhere](#)
 - [Amazon GameLift Servers Frotas Anywhere](#)
 - [Amazon GameLift Servers EC2 frotas gerenciadas](#)
 - [Amazon GameLift Servers frotas de contêineres gerenciadas](#)
- Personalize sua integração com o servidor de jogos adicionando recursos opcionais. Por exemplo, talvez você queira adicionar sessões de jogadores com um único jogador IDs, configurar o preenchimento de matchmaking ou gerenciar o acesso do servidor de jogos aos seus outros AWS recursos (como um banco de dados ou serviço de armazenamento de conteúdo). Consulte estes tópicos para obter orientação:
 - [Adicione Amazon GameLift Servers ao seu servidor de jogos](#)
 - [SDK 5.x do servidor C++ \(Unreal\) para -- Ações Amazon GameLift Servers](#)
- Personalize seu componente cliente de jogo para solicitar sessões de jogo, receber informações de conexão e conectar-se diretamente a um servidor de jogos para jogar. Consulte estes tópicos para obter orientação:

- [Integre seu cliente de jogo com Amazon GameLift Servers](#)

Amazon GameLift Servers Integre-se a um projeto Unity

Aprenda a integrar o Amazon GameLift Servers SDK for Unity em seus projetos de jogos para acessar o conjunto completo de recursos do SDK do servidor.

Dica

Para uma implantação mais rápida, experimente o plug-in Amazon GameLift Servers autônomo para Unity. Ele fornece fluxos de trabalho de interface de usuário guiados para implantar rapidamente seu servidor de jogos com configuração mínima, para que você possa testar os componentes do jogo em ação. Consulte [Amazon GameLift Servers plugin para Unity \(servidor SDK 5.x\)](#).

Recursos adicionais:

- [SDK 5.x do servidor C# para Amazon GameLift Servers -- Ações](#)
- [the section called “Obter ferramentas de desenvolvimento”](#)

Instale o SDK do servidor para Unity

Obtenha o código aberto Amazon GameLift Servers para Unity em. [GitHub](#) Os arquivos readme do repositório contêm pré-requisitos e instruções de instalação.

Configure uma frota Amazon GameLift Servers Anywhere para testes

Você pode configurar sua estação de trabalho de desenvolvimento como uma frota de hospedagem Amazon GameLift Servers Anywhere para testar iterativamente sua Amazon GameLift Servers integração. Com esta configuração, você pode iniciar os processos do servidor de jogos em sua estação de trabalho, enviar solicitações de ingresso ou matchmaking de jogadores Amazon GameLift Servers para iniciar sessões de jogo e conectar clientes às novas sessões de jogo. Com sua própria estação de trabalho configurada como servidor de hospedagem, você pode monitorar todos os aspectos da integração do seu jogo com Amazon GameLift Servers.

Para obter instruções sobre como configurar sua estação de trabalho, consulte [Configure testes locais com o Amazon GameLift Servers Anywhere](#) para concluir as seguintes etapas:

1. Crie um local personalizado para sua estação de trabalho.
2. Crie uma frota Amazon GameLift Servers em qualquer lugar com sua nova localização personalizada. Se for bem-sucedida, essa solicitação retornará um ID da frota. Anote esse valor, pois você precisará disso mais tarde.
3. Registre sua estação de trabalho como uma computação na nova frota Anywhere. Forneça um nome de computação exclusivo e especifique o endereço IP da sua estação de trabalho. Se for bem-sucedida, essa solicitação retornará um endpoint do SDK de serviço, na forma de uma WebSocket URL. Anote esse valor, pois você precisará disso mais tarde.
4. Gere um token de autenticação para a computação da sua estação de trabalho. Essa autenticação de curta duração inclui o token e uma data de expiração. Seu servidor de jogo o usa para autenticar a comunicação com o Amazon GameLift Servers serviço. Armazene a autenticação no computador da sua estação de trabalho para que os processos do servidor de jogos em execução possam acessá-la.

Adicione Amazon GameLift Servers o código do servidor ao seu projeto Unity

Seu servidor de jogo se comunica com o Amazon GameLift Servers serviço para receber instruções e relatar o status contínuo. Para fazer isso, você adiciona o código do servidor de jogos que usa o SDK Amazon GameLift Servers do servidor.

O exemplo de código fornecido ilustra os elementos básicos de integração necessários. Ele usa um `MonoBehavior` para ilustrar uma inicialização simples do servidor de jogos com Amazon GameLift Servers. O exemplo pressupõe que o servidor do jogo seja executado em uma frota Amazon GameLift Servers Anywhere para testes. Inclui código para:

- Inicialize um cliente de Amazon GameLift Servers API. A amostra usa a versão de `InitSDK()` com parâmetros de servidor para sua frota e computação do Anywhere. Use o WebSocket URL, o ID da frota, o nome do computador (ID do host) e o token de autenticação, conforme definido no tópico [Configure uma frota Amazon GameLift Servers Anywhere para testes](#) anterior.
- Implemente funções de retorno de chamada para responder às solicitações do Amazon GameLift Servers serviço `OnStartGameSession`, incluindo `OnProcessTerminate`, e. `onHealthCheck`
- Ligue para `ProcessReady()` com uma porta designada para notificar o Amazon GameLift Servers serviço quando o processo estiver pronto para hospedar sessões de jogo.

O código de amostra fornecido estabelece a comunicação com o Amazon GameLift Servers serviço. Ele também implementa um conjunto de funções de retorno de chamada que respondem às

solicitações do Amazon GameLift Servers serviço. Para obter mais informações sobre cada função e o que o código faz, consulte [Inicializar o processo do servidor](#). Para obter mais informações sobre as ações do SDK e os tipos de dados usados nesse código, leia [SDK 5.x do servidor C# para Amazon GameLift Servers -- Ações](#).

O código de exemplo mostra como adicionar a funcionalidade necessária, conforme descrito em [Adicionar Amazon GameLift Servers ao seu servidor de jogo](#). Para obter mais informações sobre as ações do SDK do servidor, consulte o. [SDK 5.x do servidor C# para Amazon GameLift Servers -- Ações](#)

Código de amostra de integração

```
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);
        }
    }
}
```

Próximas etapas

Agora que você preparou uma versão de servidor de jogos com a funcionalidade mínima necessária para hospedagem Amazon GameLift Servers, considere estas possíveis próximas etapas:

- Implante seu servidor de jogos integrado para testes e desenvolvimento. Com uma frota Anywhere, você pode configurar sua máquina local como um recurso de hospedagem e usá-la para testar as conexões do servidor e do cliente do jogo. Para hospedagem baseada em nuvem, implante seu servidor de jogos em uma frota de contêineres gerenciada EC2 ou gerenciada. Consulte estes tópicos para obter orientação:
 - [Configure para desenvolvimento iterativo com Amazon GameLift Servers Anywhere](#)
 - [Amazon GameLift Servers Frotas Anywhere](#)
 - [Amazon GameLift Servers EC2 frotas gerenciadas](#)
 - [Amazon GameLift Servers frotas de contêineres gerenciadas](#)

- Personalize sua integração com o servidor de jogos adicionando recursos opcionais. Por exemplo, talvez você queira adicionar sessões de jogadores com um único jogador IDs, configurar o preenchimento de matchmaking ou gerenciar o acesso do servidor de jogos aos seus outros AWS recursos (como um banco de dados ou serviço de armazenamento de conteúdo). Consulte estes tópicos para obter orientação:
 - [Adicione Amazon GameLift Servers ao seu servidor de jogos](#)
 - [SDK 5.x do servidor C++ \(Unreal\) para -- Ações Amazon GameLift Servers](#)
- Personalize seu componente cliente de jogo para solicitar sessões de jogo, receber informações de conexão e conectar-se diretamente a um servidor de jogos para jogar. Consulte estes tópicos para obter orientação:
 - [Integre seu cliente de jogo com Amazon GameLift Servers](#)

Projete seu serviço de cliente de jogos

Recomendamos que você implemente um serviço de cliente de jogos que autentique seus jogadores e se comunique com o Amazon GameLift Servers API. Ao implementar um serviço personalizado de cliente de jogos, você poderá:

- Personalizar a autenticação para os jogadores.
- Controle como Amazon GameLift Servers partidas e inicia sessões de jogo.
- Usar o banco de dados de jogadores para atributos de jogadores, como classificação de habilidade para marcação de jogos, em vez de confiar no cliente.

Usar um serviço de cliente de jogos também reduz os riscos de segurança introduzidos por clientes de jogos que interagem diretamente com seu Amazon GameLift Servers API.

Autenticar os jogadores

Você pode usar o Amazon Cognito e a sessão do jogador IDs para autenticar seus clientes de jogos. Para gerenciar o ciclo de vida e as propriedades das identidades de jogadores, use os grupos de usuários do Amazon Cognito.

Se preferir, crie uma solução de identidade personalizada e hospede-a na AWS. Também é possível usar os autorizadores Lambda para uma lógica de autorização personalizada com a API Gateway.

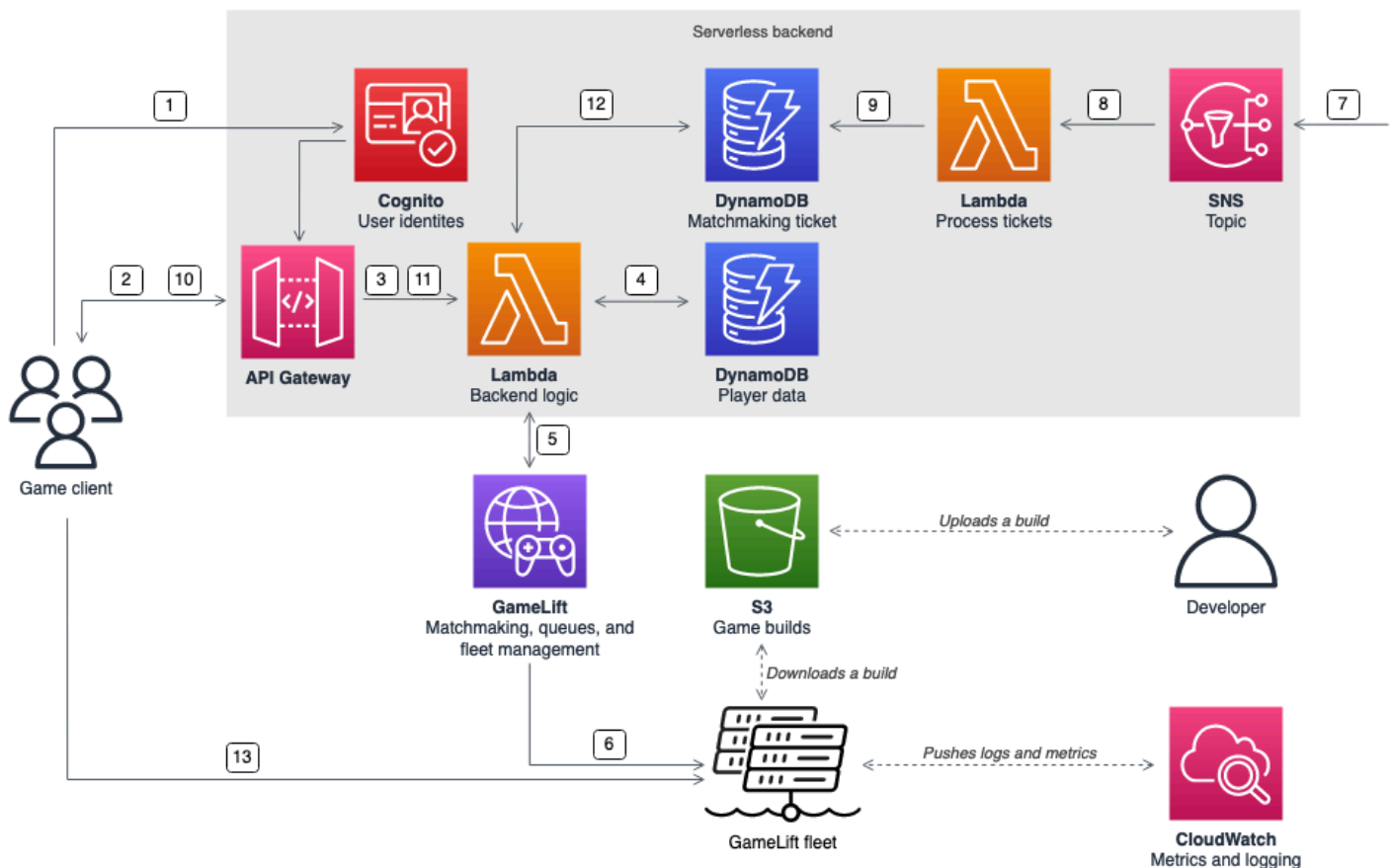
Recursos adicionais:

- [Usar banco de identidades \(identidades federadas\)](#) (Guia do desenvolvedor do Amazon Cognito)
- [Introdução aos grupos de usuários](#) (Guia do Desenvolvedor do Amazon Cognito)
- [Como configurar a autenticação do jogador com o Amazon Cognito \(AWS para blogs de jogos\)](#)

Servidores de sessão de jogo autônomos com um back-end de tecnologia sem servidor

Usando uma arquitetura de atendimento ao cliente sem servidor, o back-end pode visualizar o status dos tickets de matchmaking em um banco de dados altamente escalável, em vez de acessar diretamente o Amazon GameLift Servers API.

O diagrama a seguir mostra um back-end sem servidor criado com o Serviços da AWS qual os jogadores são colocados em jogos executados em Amazon GameLift Servers frotas. A lista a seguir fornece uma descrição para cada texto explicativo numerado no diagrama. Para experimentar este exemplo, consulte [Hospedagem de jogos baseada em sessão multijogador](#) ativada. AWS GitHub



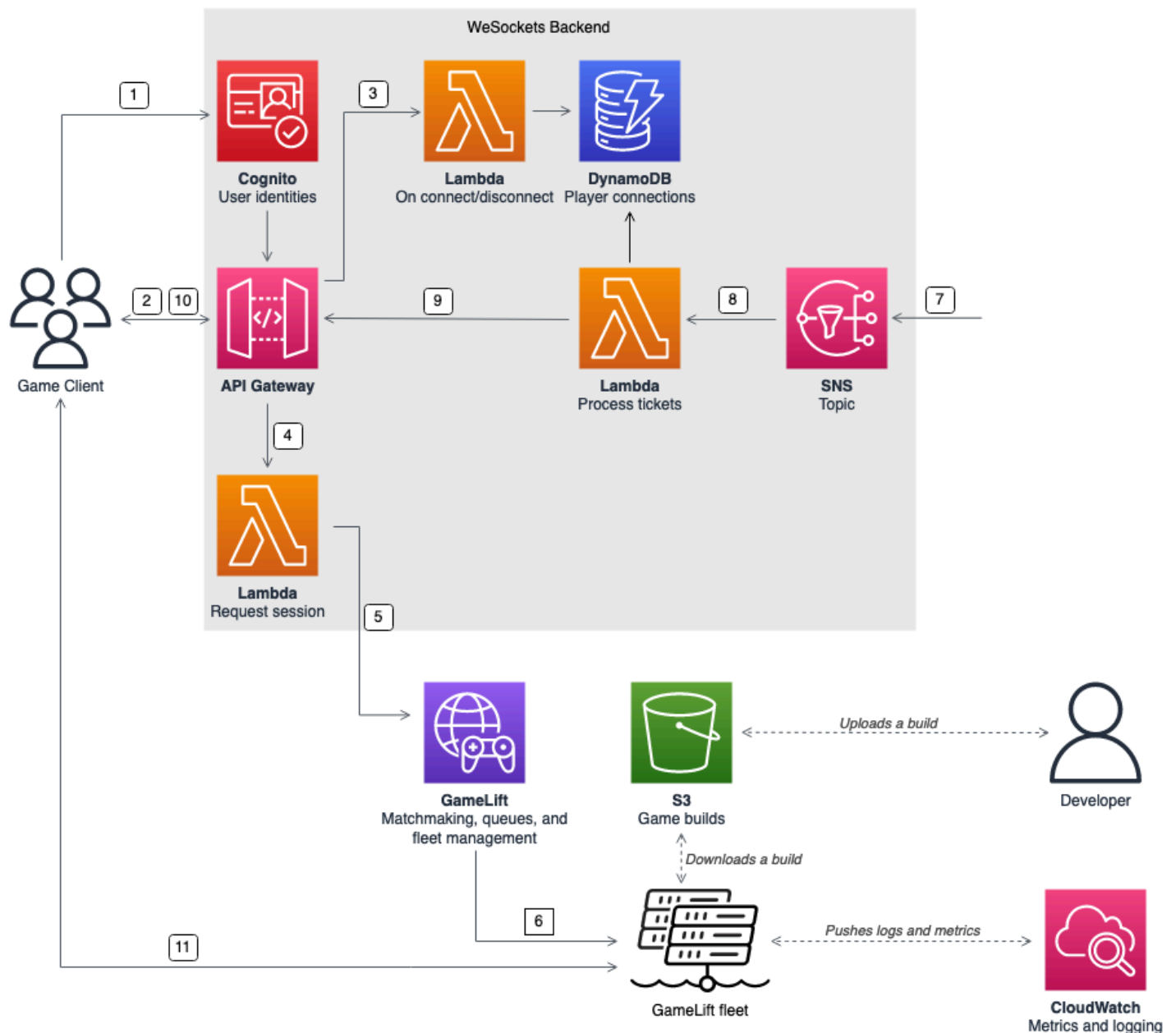
1. O cliente do jogo solicita uma identidade de usuário do Amazon Cognito a partir de um banco de identidades do Amazon Cognito.
2. O cliente do jogo recebe credenciais de acesso temporárias e solicita uma sessão de jogo por meio de uma API do Amazon API Gateway.
3. O API Gateway invoca uma AWS Lambda função.
4. A função do Lambda solicita dados do player de uma tabela NoSQL do Amazon DynamoDB. A função fornece a identidade do Amazon Cognito nos dados de contexto da solicitação.
5. A função Lambda solicita uma correspondência por meio de Amazon GameLift Servers FlexMatch matchmaking.
6. FlexMatch combina um grupo de jogadores com latência adequada e, em seguida, solicita a colocação da sessão de jogo por meio de um Amazon GameLift Servers queue. A fila tem frotas com um ou mais Região da AWS locais nela.
7. Depois Amazon GameLift Servers coloca a sessão em um dos locais da frota, Amazon GameLift Servers envia uma notificação de evento para um tópico do Amazon Simple Notification Service (Amazon SNS).
8. Uma função do Lambda recebe o evento do Amazon SNS e o processa.
9. Se o ticket de marcação de jogos for um evento de MatchmakingSucceeded, a função do Lambda grava o resultado, junto com a porta e o endereço IP do servidor do jogo, em uma tabela do DynamoDB.
10. O cliente do jogo faz uma solicitação assinada ao API Gateway para visualizar o status do ticket de marcação de jogos em um intervalo específico.
11. O API Gateway usa uma função do Lambda que verifica o status do ticket de marcação de jogos.
12. A função do Lambda verifica a tabela do DynamoDB para visualizar se o ticket foi bem-sucedido. Se for bem-sucedida, a função envia a porta e o endereço IP do servidor do jogo, junto com o ID da sessão do jogador, de volta ao cliente. Se o ticket não for bem-sucedido, a função enviará uma resposta verificando se a partida ainda não está pronta.
13. O cliente do jogo se conecta ao servidor do jogo usando TCP ou UDP usando a porta e o endereço IP fornecidos pelo serviço de back-end. O cliente do jogo então envia o ID da sessão do jogador para o servidor do jogo, que então valida o ID usando o SDK do servidor para Amazon GameLift Servers.

Servidores de sessão de jogo autônomos com um back-end WebSocket baseado

Usando uma arquitetura WebSocket baseada no Amazon API Gateway, você pode fazer solicitações de matchmaking WebSockets e enviar notificações push para conclusão de matchmaking usando mensagens iniciadas pelo servidor. Essa arquitetura melhora o desempenho por meio da comunicação bidirecional entre o cliente e o servidor.

Para obter mais informações sobre como usar o API Gateway WebSock APIs, consulte [Trabalhando com WebSocket APIs](#).

O diagrama a seguir mostra uma arquitetura de back-end WebSocket baseada que usa o API Gateway e outros Serviços da AWS para combinar jogadores em jogos executados em Amazon GameLift Servers frotas. A lista a seguir fornece uma descrição para cada texto explicativo numerado no diagrama.



1. O cliente do jogo solicita uma identidade de usuário do Amazon Cognito a partir de um banco de identidades do Amazon Cognito.
2. O cliente do jogo assina uma WebSocket conexão com uma API do API Gateway com as credenciais do Amazon Cognito.
3. O API Gateway chama uma AWS Lambda função na conexão. A função armazena as informações de conexão em uma tabela do Amazon DynamoDB.
4. O cliente do jogo envia uma mensagem para uma função Lambda, por meio da API API Gateway pela WebSocket conexão, para solicitar uma sessão.

5. Uma função Lambda recebe a mensagem e, em seguida, solicita uma correspondência por meio de Amazon GameLift Servers FlexMatch matchmaking.
6. Depois FlexMatch combina com um grupo de jogadores, FlexMatch solicita a colocação de uma sessão de jogo por meio de um Amazon GameLift Servers queue.
7. Depois Amazon GameLift Servers coloca a sessão em um dos locais da frota, Amazon GameLift Servers envia uma notificação de evento para um tópico do Amazon Simple Notification Service (Amazon SNS).
8. Uma função do Lambda recebe o evento do Amazon SNS e o processa.
9. Se o ticket de marcação de jogos for um evento de MatchmakingSucceeded, a função do Lambda solicitará a conexão correta do jogador do DynamoDB. Em seguida, a função envia uma mensagem para o cliente do jogo por meio da API Gateway API pela WebSocket conexão. Nessa arquitetura, o cliente do jogo não pesquisa ativamente o status de marcação de jogos.
10. O cliente do jogo recebe a porta e o endereço IP do servidor do jogo, junto com o ID da sessão do jogador, por meio da WebSocket conexão.
11. O cliente do jogo se conecta ao servidor do jogo usando TCP ou UDP usando a porta e o endereço IP fornecidos pelo serviço de back-end. O cliente do jogo também envia o ID da sessão do jogador para o servidor do jogo, que então valida o ID usando o SDK do servidor para Amazon GameLift Servers.

Configure para desenvolvimento iterativo com Amazon GameLift Servers Anywhere

Amazon GameLift Servers fornece ferramentas e soluções para ajudá-lo a configurar um ambiente de teste hospedado para uso durante o desenvolvimento do jogo. Com essas ferramentas, você pode criar um ambiente que espelhe a experiência real do jogador com hospedagem gerenciada com Amazon GameLift Servers e oferece suporte a um processo de desenvolvimento rápido e iterativo.

Com um ambiente de teste separado, você remove a sobrecarga de um Amazon GameLift Servers frota gerenciada durante os testes. Não é mais necessário fazer upload de cada nova iteração de compilação do servidor de jogos, criar uma frota para ela e esperar mais de 15 minutos para que seja ativada. Em vez disso, é possível criar uma compilação, atualizar rapidamente a frota de testes com a nova compilação, iniciá-la e começar os testes.

Usando um Amazon GameLift Servers Em qualquer frota, você pode configurar um ambiente de teste usando um dispositivo local, como sua estação de trabalho de desenvolvimento. Você também pode configurar um ambiente de teste usando um recurso de hospedagem baseado em nuvem.

Configure um ambiente de teste Anywhere para desenvolver e testar uma variedade de cenários, incluindo estes:

- Teste a integração do seu servidor de jogos com o Amazon GameLift Servers SDK do servidor. Você pode testar mesmo sem um cliente de jogo em funcionamento usando chamadas de AWS CLI para iniciar novas sessões de jogo e acompanhar eventos de sessões de jogos.
- Teste as interações entre o cliente do jogo, o serviço de back-end e o Amazon GameLift Servers serviço à medida que você desenvolve componentes para seu jogo. Ajuste a experiência do jogador ao ingressar em um jogo.
- Experimente com o seu FlexMatch design de matchmaker. Teste as variações de conjuntos de regras e outras implementações de recursos de criação de parcerias. Configure e teste o preenchimento de criação de parcerias.
- Experimente outros Amazon GameLift Servers recursos de hospedagem, como configurações de tempo de execução (com o Amazon GameLift Servers Agente) para gerenciamento do ciclo de vida do servidor de jogos.
- Crie, teste e repita rapidamente para validar todos os aspectos da experiência do jogador no jogo, incluindo interações multijogador, em um ambiente hospedado ao vivo.

Posteriormente, ao preparar seu jogo para o lançamento, você desejará adicionar Amazon GameLift Servers frotas gerenciadas para ajustar suas configurações de hospedagem e testar cenários adicionais, incluindo os seguintes:

- Experimente e teste designs de filas de sessões de jogos, incluindo o uso de frotas com vários locais, frotas spot e sob demanda e vários tipos de instâncias.
- Teste as opções de posicionamento de sessões de jogos com frotas gerenciadas, incluindo o uso de políticas opcionais de latência e configurações de priorização da frota.
- Configure o ajuste de escala de capacidade para atender à demanda dos jogadores, usando opções de ajuste de escala automático ou manual.
- Configurar AWS CloudFormation com Amazon GameLift Servers frotas gerenciadas para gerenciar seus recursos de hospedagem a longo prazo.

Fast Build Update Tool (somente para desenvolvimento)

Com EC2 frotas gerenciadas, para implantar uma atualização de compilação do servidor de jogos, você precisa fazer o upload de cada nova compilação para Amazon GameLift Servers e crie uma nova frota para ele.

A Fast Build Update Tool permite que você ignore essas etapas durante o desenvolvimento, economizando tempo e permitindo uma iteração de desenvolvimento mais rápida. Com essa ferramenta, você pode atualizar rapidamente os arquivos de compilação de jogos em todos os computadores de uma frota existente. A ferramenta tem várias opções: você pode substituir uma versão inteira do jogo ou alterar 6 arquivos específicos e gerenciar como reiniciar os processos do servidor de jogos após as atualizações. Ela também permite atualizar computadores individuais em uma frota.

Para obter a Fast Build Update Tool e saber mais sobre como usá-la, visite o Amazon GameLift Servers Repositório do kit de ferramentas para [a ferramenta Fast Build Update](#) no Github.

Tópicos

- [Criar um ambiente de teste baseado em nuvem](#)
- [Configure testes locais com o Amazon GameLift Servers Anywhere](#)
- [Trabalhe com o Amazon GameLift Servers agente](#)

Criar um ambiente de teste baseado em nuvem

Note

Este tópico aborda testes iterativos para jogos integrados ao SDK do servidor para Amazon GameLift Servers versão 5.x. Se o jogo usa o SDK do servidor versão 4.x ou anterior, consulte [Teste sua integração usando Amazon GameLift Servers Local](#).

Use um Amazon GameLift Servers Frota em qualquer lugar para criar e testar de forma iterativa os componentes do seu jogo em um ambiente hospedado na nuvem. Crie uma frota em qualquer lugar com recursos de hospedagem e uma conexão com o Amazon GameLift Servers serviço, execute seus servidores de jogos neles e teste a funcionalidade do jogo conforme necessário.

Implante uma frota em qualquer lugar com o Amazon GameLift Servers Agente

Se a versão do seu servidor de jogos estiver integrada com Amazon GameLift Servers SDK 5.x ou posterior, você pode implantá-lo em uma frota Anywhere baseada em nuvem com o Amazon GameLift Servers Agente. O Agent é um processo em segundo plano que gerencia os ciclos de vida do servidor de jogos e outras tarefas em cada unidade de computação de uma frota. Essas tarefas incluem registrar a computação em uma frota do Anywhere, adquirir um token de autenticação e iniciar/interromper processos do servidor de jogos com base em um conjunto de instruções. O Agent é controlado pela configuração de tempo de execução de uma frota, que você pode atualizar a qualquer momento durante o ciclo de vida da frota. (O agente é implantado automaticamente em EC2 frotas gerenciadas.) Para obter mais informações e fazer o download do Agente, consulte [Amazon GameLift Servers GitHubrepositório](#).

Configure testes iterativos com a Amazon EC2

Use o fluxo de trabalho guiado neste [Amazon GameLift Servers solução de kit de ferramentas](#) para configurar um ambiente de hospedagem baseado em nuvem que espelhe a experiência de hospedagem gerenciada com Amazon GameLift Servers.

O GitHub repositório fornece um conjunto de scripts que automatizam a maioria dos processos para configurar um ambiente de teste com Amazon GameLift Servers Em qualquer lugar e no Amazon GameLift Servers Agente. Ele também fornece orientação para atualizar o ambiente sempre que você tiver uma nova versão do servidor de jogos para testar. Você pode executar um único script que implanta um ambiente de teste com um exemplo de compilação de servidor de jogos ou seguir cada etapa para configurá-lo com sua própria compilação de servidor de jogos.

Nesse fluxo de trabalho, você trabalhará inteiramente no AWS Management Console, usando AWS CloudShell para executar scripts e concluir tarefas de linha de comando.

Note

Para as tarefas deste tutorial, você precisa de um usuário AWS da conta com permissões para os seguintes serviços: Amazon GameLift Servers AWS CloudShell, Amazon S3 EC2, AWS Systems Manager Amazon e. AWS Identity and Access Management Os usuários com acesso administrativo à AWS conta já têm as permissões necessárias.

O fluxo de trabalho abrange as seguintes tarefas:

- Package um servidor de jogo construído para Amazon GameLift Servers. O fluxo de trabalho fornece um script para criar um servidor de jogos C++ de amostra, que já foi integrado ao SDK do servidor para Amazon GameLift Servers versão 5.x e está pronto para hospedagem. Como alternativa, você pode trabalhar com seu próprio projeto de jogo se tiver concluído a integração.
- Configure um bucket do Amazon Simple Storage Service para armazenar compilações e dependências de servidores de jogos. Ao produzir novas versões de suas compilações de jogos, você pode armazená-las no S3 e usar os scripts para atualizar a frota Anywhere para testes de jogos.
- Obtenha e construa o Amazon GameLift Servers Agente. O Agent gerencia os processos do servidor de jogos em um recurso de hospedagem com base na configuração. Ele usa a mesma lógica e se comporta de forma idêntica a Amazon GameLift Servers EC2 hospedagem gerenciada.
- Configure uma frota Anywhere para seus recursos de hospedagem. Com uma frota Anywhere, você pode usar o Amazon GameLift Servers serviço para hospedagem de recursos que não são gerenciados pelo Amazon GameLift Servers. Nesta etapa, você também configurará a configuração do tempo de execução, que instrui Amazon GameLift Servers Agente quando e como iniciar os processos do servidor de jogos.
- Configure uma EC2 instância da Amazon. Esse é o seu ambiente de teste para testes iterativos. É muito mais rápido usar uma EC2 instância padrão em vez de uma totalmente gerenciada Amazon GameLift Servers instância (que é otimizada para uso em nível de produção). Com uma EC2 instância padrão, você pode atualizar o servidor do jogo de forma rápida e contínua, conforme necessário.
- Implante seu servidor de jogos, construa e Amazon GameLift Servers Agente para a EC2 instância da Amazon. O fluxo de trabalho fornece um script que obtém a versão mais recente da compilação do seu jogo e todas as dependências e a instala na sua EC2 instância. Nesse fluxo de trabalho, as dependências incluem Amazon GameLift Servers Agente e o CloudWatch agente.
- Inicie o Amazon GameLift Servers Agente. Depois de instalado, o Agent inicia e começa a executar as instruções automaticamente. Isso inclui:
 - Registre a EC2 instância como uma computação no Amazon GameLift Servers Frota em qualquer lugar.
 - Estabeleça uma WebSocket conexão com o Amazon GameLift Servers faça o serviço e obtenha a configuração de tempo de execução mais recente.
 - Inicie os processos do servidor de jogos com base nas instruções na configuração do tempo de execução. Nesse fluxo de trabalho, o Agent é instruído a iniciar um único processo do executável do servidor de jogos.

- Teste os cenários de jogo. Com o ambiente de teste configurado e a versão mais recente do servidor de jogos instalada, você pode começar os testes. O fluxo de trabalho percorre várias etapas de teste, incluindo o início de uma sessão de jogo. Acesse os registros do servidor do CloudWatch jogo para acompanhar o progresso à medida que a sessão do jogo começa e se prepara para aceitar jogadores.

Ao desenvolver os componentes do jogo, incluindo um cliente de jogo e um serviço de backend do lado do cliente, você pode incluí-los nos cenários de teste. Use um cliente de jogo para solicitar uma sessão de jogo, recuperar informações de conexão do Amazon GameLift Servers serviço e, em seguida, conecte-se diretamente à sessão do jogo.

- Implante uma nova compilação de servidor de jogos e repita os testes. Ao desenvolver seu jogo, você pode gerar novas compilações de servidores de jogos e, em seguida, implantá-las rapidamente no ambiente de EC2 teste para testes. Faça upload delas para o bucket do Amazon S3 e use os scripts de fluxo de trabalho para atualizar o ambiente de teste.

Faça a transição do seu jogo para Amazon GameLift Servers frotas gerenciadas

Depois de concluir os testes de desenvolvimento e estar pronto para se preparar para o lançamento, é um bom momento para mudar para Amazon GameLift Servers frotas gerenciadas. Use frotas gerenciadas para ajustar e testar os recursos de hospedagem de jogos. Implemente a solução de posicionamento de sessões de jogo (filas e matchmakers), selecione o hardware de hospedagem ideal (incluindo frotas spot) e os locais e escolha uma estratégia para escalar a capacidade. Talvez você também queira começar a usar AWS CloudFormation para gerenciar com mais eficiência os ciclos de vida de todos os seus recursos de hospedagem de jogos, incluindo frotas, filas e matchmakers.

É necessário um esforço mínimo para fazer a transição de uma frota de testes Anywhere baseada em nuvem para uma Amazon GameLift Servers frota gerenciada. Não é necessário alterar nenhum código de jogo, e você pode reutilizar as mesmas filas e matchmakers. Realize as seguintes tarefas:

- Crie um Amazon GameLift Servers recurso de construção. Com uma frota de testes Anywhere, você precisa implantar manualmente a compilação e as dependências do servidor de jogos em cada computação da frota. Com uma frota gerenciada, faça o upload do pacote de compilação do jogo para Amazon GameLift Servers, que o implanta automaticamente em todos os computadores da frota. Consulte [Implemente uma criação de servidor personalizada para Amazon GameLift Servers hospedagem](#) para obter detalhes sobre como empacotar os arquivos de compilação de jogos e criar um recurso de compilação com arquivos em um bucket do Amazon S3.

- Crie uma frota gerenciada. Crie uma frota usando o console ou a AWS CLI, especificando uma EC2 frota gerenciada. Esse tipo de frota exige configurações adicionais, incluindo a especificação do recurso de compilação e dos tipos de instância. Você pode usar a mesma configuração de tempo de execução para gerenciar o ciclo de vida do servidor de jogos em cada computação da frota. Consulte [Crie uma EC2 frota Amazon GameLift Servers gerenciada](#) para obter detalhes sobre como criar uma frota gerenciada.
- Redirecione os aliases da frota (opcional). Se você configurar aliases para usar com suas frotas Anywhere, poderá reutilizar os mesmos aliases para suas frotas gerenciadas. Consulte [Crie um Amazon GameLift Servers alias](#) para obter detalhes sobre como criar ou atualizar um alias.

Configure testes locais com o Amazon GameLift Servers Anywhere

Note

Este tópico aborda testes locais para jogos integrados ao SDK do servidor para a Amazon GameLift Servers versão 5.x. Se o jogo usa o SDK do servidor versão 4.x ou anterior, consulte [Teste sua integração usando Amazon GameLift Servers Local](#).

Use uma frota Amazon GameLift Servers Anywhere e seu próprio hardware para criar e testar iterativamente seus componentes de jogo em um ambiente hospedado simulado. Configure uma frota Anywhere e registre um dispositivo local para estabelecer uma conexão com o Amazon GameLift Servers serviço. Instale a compilação do servidor de jogos no dispositivo, inicie um processo de servidor de jogos e teste a funcionalidade do jogo conforme necessário. Você pode atualizar a compilação do servidor de jogos sempre que necessário para testar cada nova iteração de compilação.

Com uma frota Anywhere, você pode testar usando a AWS CLI ou com scripts de teste. Se você integrou um cliente de jogo com Amazon GameLift Servers, você pode executar o cliente no mesmo dispositivo local ou em um dispositivo diferente.

Testar localmente com uma frota Anywhere é particularmente útil para testar a integração do servidor de jogos com Amazon GameLift Servers. Você tem visibilidade total de todas as atividades de hospedagem na máquina local, bem como de eventos e dados de registro em log.

 Note

Você está usando o Amazon GameLift Servers plugin para Unreal Engine ou Unity? Essas ferramentas incluem fluxos de trabalho guiados para configurar testes locais com uma frota Anywhere. Siga a documentação para [Plugin para Unity: configure testes locais com Amazon GameLift Servers Anywhere](#) ou [Plugin para Unreal: hospede seu jogo localmente com Anywhere Amazon GameLift Servers](#).

Tópicos

- [Configure uma frota local Anywhere](#)
- [Atualizar e instalar o servidor de jogos](#)
- [Testar atividades de sessões de jogo](#)
- [Faça iterações em seu servidor de jogos](#)
- [Faça a transição do seu jogo para frotas Amazon GameLift Servers gerenciadas](#)

Configure uma frota local Anywhere

Siga estas etapas para criar uma frota Anywhere para sua estação de trabalho local. Para obter instruções detalhadas sobre o uso da AWS CLI ou do AWS Management Console for Amazon GameLift Servers, consulte [Crie um Amazon GameLift Servers Frota em qualquer lugar](#)

Para criar a frota Anywhere

1. Crie um local personalizado para a estação de trabalho local. (AWS CLI ou console). Um local personalizado é simplesmente um rótulo para o recurso computacional que você planeja incluir na sua frota Anywhere. Os nomes de locais personalizados devem começar com custom-. Por exemplo: custom-my_laptop. Consulte [Criar uma política personalizada](#).
2. Crie uma frota Anywhere (AWS CLI ou console). Nesta etapa, crie o recurso de frota com o local personalizado para a estação de trabalho local. Consulte [Criar uma frota Anywhere](#).

Anote o valor do ID ou ARN da nova frota. Você precisará desse valor na próxima etapa.

3. Registre sua estação de trabalho local como uma computação de frota (somente AWS CLI). Uma frota Anywhere deve ter pelo menos um recurso computacional para hospedar seus servidores de jogos. Consulte [Adicionar um recurso de computação à frota](#). Para adicionar uma computação à frota, você precisa das seguintes informações:

- Um nome de recurso de computação. Cada recurso de computação em uma frota deve ter um nome exclusivo.
- O identificador de frota Anywhere. Você pode usar o `FleetID` ou o `FleetArn`.
- As informações de conexão do recurso de computação. Especifique um `IpAddress` ou `DnsName`. É assim que os clientes Amazon GameLift Servers de jogos se conectarão aos servidores de jogos.
- Um local personalizado na frota Anywhere.

Anote o valor de retorno do `GameLiftServiceSdkEndpoint`. Você precisará desse valor ao atualizar seu servidor de jogo para ser executado em uma frota Anywhere.

Atualizar e instalar o servidor de jogos

Essa tarefa pressupõe que você já tenha integrado uma compilação de servidor de jogos com o SDK Amazon GameLift Servers 5.x do servidor. O processo de integração envolve a adição de código ao servidor do jogo para que ele possa interagir com o Amazon GameLift Servers serviço para iniciar e gerenciar as sessões do jogo.

Para uma frota Anywhere, você precisa definir manualmente determinadas configurações do servidor do jogo. Em uma frota Amazon GameLift Servers gerenciada, essas configurações são definidas automaticamente.

Para preparar seu servidor de jogo para uma frota Anywhere

1. Obtenha um token de autenticação. Seu servidor de jogo deve incluir um token de autenticação em cada comunicação com o Amazon GameLift Servers serviço. Amazon GameLift Servers tokens de autenticação duram pouco e devem ser atualizados regularmente.

Como prática recomendada, crie um script para concluir as seguintes tarefas:

- Chame a ação da AWS CLI. `get-compute-auth-token`
- Armazene o valor do token retornado onde os processos do servidor de jogos possam recuperá-lo, como em uma variável de ambiente na computação local.

Instale o script com o servidor de jogos no recurso de computação. Defina o script para ser executado antes de iniciar o primeiro processo do servidor de jogos. Enquanto os processos

do servidor de jogos estiverem ativos, execute o script regularmente para manter um token de autenticação válido. Todos os processos do servidor de jogos no recurso de computação podem usar o mesmo token de autenticação.

2. Atualize o código do servidor do Amazon GameLift Servers jogo. Ao integrar o código do servidor do jogo ao SDK do servidor Amazon GameLift Servers, você adicionou uma chamada à ação `InitSdk()`. Quando o servidor do jogo é executado em uma frota Anywhere, essa chamada requer parâmetros adicionais do servidor. Para obter mais informações, consulte [Inicializar o processo do servidor](#) e [SDK de servidor 5.x para Amazon GameLift Servers](#) para a linguagem de desenvolvimento. Os parâmetros do servidor são:

- `websocketUrl`: defina esse parâmetro para o valor `GameLiftServiceSdkEndpoint`, que é retornado quando você registra uma computação na frota.
- `hostId`: defina esse parâmetro com o nome da computação especificada ao registrar uma computação na frota do Anywhere.
- `fleetId`— Defina esse parâmetro como o ID da frota Anywhere.
- `authToken`: defina esse parâmetro como o token, que é retornado em resposta a uma solicitação para recuperar um token de autenticação para uma computação.
- `processId`: defina esse parâmetro para identificar um processo do servidor de jogos que está sendo executado na computação local. Cada processo simultâneo do servidor de jogos deve ter um ID de processo exclusivo.

Os valores dos parâmetros do servidor que cada processo do servidor de jogos usa precisam ser específicos para a computação da frota do Anywhere em que o processo está sendo executado. Para obter detalhes sobre como obter os valores apropriados para um cálculo, consulte [Adicionar um recurso de computação à frota](#). Como prática recomendada, defina `websocketUrl`, `hostId`, `fleetId` e `authToken` como variáveis de ambiente na computação local. Todos os processos do servidor executados na computação usarão esses valores.

3. Instale a versão do servidor de jogos na computação local. Inclua todas as dependências necessárias para executar o servidor de jogos.
4. Inicie um ou mais processos do servidor de jogos em execução na computação local. Quando o processo do servidor de jogos chama a ação `ProcessReady()` do SDK do servidor, o processo está pronto para hospedar uma sessão de jogo.

Testar atividades de sessões de jogo

Teste a integração do servidor de jogos trabalhando com sessões de jogo. Se você não tiver um cliente de jogo integrado com a Amazon GameLift Servers funcionalidade, poderá usar a AWS CLI para iniciar as sessões de jogo. Teste os seguintes cenários:

- Crie uma sessão de jogo. [create-game-session](#) Comando de chamada (ou a operação [CreateGameSession](#) da API). Especifique o ID e a localização personalizada da sua frota Anywhere. Essa chamada retorna um identificador exclusivo para a nova sessão de jogo.
- Verifique o status da sessão do jogo. [describe-game-sessions](#) Comando de chamada (ou a ação [DescribeGameSessions](#) da API). Especifique o ID da sessão do jogo. Essa chamada retorna informações detalhadas da sessão do jogo, incluindo seu status. As sessões de jogo no status Ativo estão prontas para os jogadores se conectarem. Para obter uma lista de todas as sessões de jogo da frota, chame o [list-game-sessions](#) comando (ou a ação da [ListGameSessions](#) API).
- Conecte-se à sessão do jogo. Se o cliente tiver a capacidade de entrar em uma sessão de jogo, use as informações de conexão incluídas nas informações da sessão do jogo.

Faça iterações em seu servidor de jogos

Você pode usar a mesma frota Anywhere e a mesma computação para testar outras versões da versão do servidor de jogos.

1. Apague a **GameSession** existente. Se o processo do servidor do jogo falhar ou não `ligarProcessEnding()`, Amazon GameLift Servers limpe-o `GameSession` depois que o servidor do jogo parar de enviar verificações de saúde.
2. Gere uma nova compilação de servidor de jogos. Faça alterações no servidor de jogos e empacote uma versão revisada.
3. Atualize a compilação do servidor de jogos na computação local. Sua frota anterior do Anywhere ainda está ativa e seu laptop ainda está registrado como um recurso computacional na frota.
4. Obtenha um token de autorização atualizado. Chame o comando da [get-compute-auth-token](#) CLI e armazene o token na computação local.
5. Inicie um ou mais processos do servidor de jogos em execução na computação local. Quando o processo do servidor de jogos chama `ProcessReady()`, ele está pronto para ser usado para testes.

Faça a transição do seu jogo para frotas Amazon GameLift Servers gerenciadas

Depois de concluir os testes de desenvolvimento e estar pronto para se preparar para o lançamento, esse é um bom momento para mudar para frotas Amazon GameLift Servers gerenciadas. Use frotas gerenciadas para ajustar e testar os recursos de hospedagem de jogos. Implemente a solução de posicionamento de sessões de jogo (filas e matchmakers), selecione o hardware de hospedagem ideal (incluindo frotas spot) e os locais e escolha uma estratégia para escalar a capacidade. Talvez você também queira começar a usar AWS CloudFormation para gerenciar com mais eficiência os ciclos de vida de todos os seus recursos de hospedagem de jogos, incluindo frotas, filas e matchmakers.

Você precisa fazer algumas pequenas modificações para fazer a transição de uma frota de teste local do Anywhere para uma frota Amazon GameLift Servers gerenciada. Você pode reutilizar as mesmas filas e matchmakers. Realize as seguintes tarefas:

- Altere a chamada de código do servidor de jogos para **InitSdk()**. Remova os parâmetros do servidor. Para uma frota gerenciada, rastreia Amazon GameLift Servers automaticamente essas informações.
- Crie um recurso de Amazon GameLift Servers construção. Com uma frota de testes Anywhere, você precisa implantar manualmente a compilação e as dependências do servidor de jogos em cada computação da frota. Com uma frota gerenciada, você cria e carrega seu pacote de compilação do jogo para Amazon GameLift Servers, que o implanta automaticamente em todos os computadores da frota. Consulte [Implemente uma criação de servidor personalizada para Amazon GameLift Servers hospedagem](#) para obter detalhes sobre como empacotar os arquivos de compilação de jogos e criar um recurso de compilação com arquivos em um bucket do Amazon S3. Não inclua scripts que registrem uma computação e obtenham um token de autenticação, pois processa Amazon GameLift Servers automaticamente essas tarefas com frotas gerenciadas.
- Crie uma frota gerenciada. Crie uma frota usando o console ou a AWS CLI, especificando uma EC2 frota gerenciada. Esse tipo de frota exige configurações adicionais, incluindo a especificação do recurso de compilação e dos tipos de instância. Você também precisa definir uma configuração de tempo de execução para gerenciar o ciclo de vida do servidor de jogos em cada computação da frota. Consulte [Crie uma EC2 frota Amazon GameLift Servers gerenciada](#) para obter detalhes sobre como criar uma frota gerenciada.
- Redirecione os aliases da frota (opcional). Se você configurar aliases para usar com suas frotas Anywhere, poderá reutilizar os mesmos aliases para suas frotas gerenciadas. Consulte [Crie um Amazon GameLift Servers alias](#) para obter detalhes sobre como criar ou atualizar um alias.

Trabalhe com o Amazon GameLift Servers agente

O Amazon GameLift Servers agente supervisiona a execução dos processos do servidor de jogos em suas Amazon GameLift Servers frotas. Implantado em cada computação de uma frota, o Agent fornece gerenciamento automatizado de processos, gerenciamento de hospedagem e registro em log para a computação. Para usar o Agente, você deve ter sua compilação de servidor de jogos integrada ao SDK do servidor para a Amazon GameLift Servers versão 5.x ou posterior.

O Amazon GameLift Servers Agente está disponível externamente para uso com Amazon GameLift Servers frotas que não são frotas gerenciadas EC2. (EC2 As frotas gerenciadas lidam com as tarefas do agente automaticamente.) Você pode optar por operar Amazon GameLift Servers frotas, incluindo frotas em qualquer lugar, com ou sem o agente. Sem o Agent, será necessário uma solução alternativa para realizar as tarefas necessárias.

Quando implantado em um computador, o Amazon GameLift Servers Agente deve ser iniciado antes que qualquer processo do servidor do jogo seja iniciado. No início, o Agent realiza as seguintes tarefas:

- Registra a computação com uma frota Amazon GameLift Servers Anywhere usando a [RegisterComputeAPI](#).
- Chama a [GetComputeAuthTokenAPI](#) para buscar um token de autorização e o armazena para uso pelos processos do servidor que estão sendo executados na computação.
- Define a variável de ambiente de WebSocket URL para a computação e estabelece uma WebSocket conexão com o Amazon GameLift Servers serviço.
- Solicita a versão mais recente da configuração de tempo de execução da frota do Amazon GameLift Servers serviço.
- Inicia e interrompe os processos do servidor de acordo com as instruções de configuração de tempo de execução.

O código-fonte e as instruções de criação do Amazon GameLift Servers Agente estão disponíveis no [Amazon GameLift ServersAgente](#) GitHub.

Sobre o Agent

O Amazon GameLift Servers Agente foi projetado para lidar com as seguintes tarefas para suas frotas:

Gerenciamento de processos

- Inicia novos processos do servidor conforme definido nas instruções de tempo de execução. O Agent pode usar uma configuração de tempo de execução personalizada que é implantada com o Agent. Como alternativa, é possível fornecer uma `RuntimeConfiguration` como parte da definição da frota. Essa abordagem tem a vantagem de permitir a modificação da configuração de tempo de execução da frota a qualquer momento. O Agente solicita periodicamente configurações de tempo de execução atualizadas do Amazon GameLift Servers serviço.
- Monitora as ativações dos processos do servidor e encerra os processos quando eles não são ativados a tempo.
- Envia batimentos cardíacos para. Amazon GameLift Servers Se o Agent não enviar heartbeats, a computação poderá ser marcada como obsoleta.
- Relata Amazon GameLift Servers quando um processo do servidor termina. Amazon GameLift Servers usa essas informações para monitorar a disponibilidade do servidor de jogos para o posicionamento da sessão de jogo.
- Emite eventos de frota para processos de servidor, incluindo:
 - `SERVER_PROCESS_INVALID_PATH`: os parâmetros de inicialização do processo do servidor de jogos foram configurados incorretamente.
 - `SERVER_PROCESS_TERMINATED_UNHEALTHY`: o processo do servidor de jogos não relatou uma verificação de integridade válida dentro de 3 minutos após a ativação e, portanto, foi encerrado.
 - `SERVER_PROCESS_FORCE_TERMINATED`: o processo do servidor de jogos não foi encerrado corretamente após o envio de `OnProcessTerminate()` dentro de 30 segundos.
 - `SERVER_PROCESS_CRASHED`: um processo do servidor de jogos travou por algum motivo.

Gerenciamento de computação

- Recebe mensagens do Amazon GameLift Servers serviço para desligar a computação.
- Solicita que a computação seja encerrada por. Amazon GameLift Servers

Registro em log

- Carrega registros em um bucket do Amazon S3 em AWS sua conta.

Teste sua integração usando Amazon GameLift Servers Local

Note

Este tópico aborda os testes de jogos integrados ao SDK do servidor para Amazon GameLift Servers somente na versão 4.x ou anterior. O pacote SDK do seu servidor inclui uma versão compatível do Amazon GameLift Servers Local. Se você estiver usando o SDK do servidor versão 5.x, consulte [Configure testes locais com o Amazon GameLift Servers Anywhere](#) os testes locais com um Amazon GameLift Servers Frota em qualquer lugar.

Use Amazon GameLift Servers Local para executar uma versão limitada do gerenciado Amazon GameLift Servers serviço em um dispositivo local e teste sua integração de jogos com ele. Esta ferramenta é útil ao fazer o desenvolvimento iterativo na integração do jogo. A alternativa: fazer o upload de cada nova compilação para Amazon GameLift Servers e configurar uma frota para hospedar seu jogo — pode levar várias ou mais de cada vez.

With Amazon GameLift Servers Local, você pode verificar o seguinte:

- Seu servidor de jogo está integrado corretamente com o SDK do servidor e está se comunicando adequadamente com o Amazon GameLift Servers serviço para iniciar novas sessões de jogo, aceitar novos jogadores e relatar saúde e status.
- Seu cliente de jogo está integrado corretamente com o AWS SDK para Amazon GameLift Servers e é capaz de recuperar informações sobre sessões de jogo existentes, iniciar novas sessões de jogo, juntar jogadores aos jogos e conectar-se à sessão do jogo.

Amazon GameLift Servers Local é uma ferramenta de linha de comando que inicia uma versão independente do gerenciado Amazon GameLift Servers serviço. Amazon GameLift Servers O Local também fornece um registro de eventos em execução da inicialização do processo do servidor, das verificações de integridade e das chamadas e respostas da API. Amazon GameLift Servers O Local reconhece um subconjunto das ações do AWS SDK para Amazon GameLift Servers. Você pode fazer chamadas do AWS CLI ou do seu cliente de jogo. Todas as ações da API são executadas localmente da mesma forma que no Amazon GameLift Servers serviço web.

Cada processo do servidor deve hospedar apenas uma única sessão de jogo. A sessão do jogo é o executável que você usa para se conectar Amazon GameLift Servers Local. Quando a sessão do jogo estiver concluída, você deverá chamar `GameLiftServerSDK::ProcessEnding` e sair

do processo. Ao testar localmente com Amazon GameLift Servers Local, você pode iniciar vários processos do servidor. Cada processo se conectará a Amazon GameLift Servers Local. Em seguida, você pode criar uma sessão de jogo para cada processo do servidor. Quando sua sessão de jogo terminar, o processo do servidor de jogos deverá sair. Em seguida, você deve iniciar manualmente outro processo do servidor.

Amazon GameLift Servers local suporta o seguinte APIs:

- CreateGameSession
- CreatePlayerSession
- CreatePlayerSessions
- DescribeGameSessions
- DescribePlayerSessions

Configurar Amazon GameLift Servers local

Amazon GameLift Servers O local é fornecido como um `.jar` arquivo executável incluído no SDK do [servidor](#). Ele pode ser executado no Windows ou Linux e usado com qualquer Amazon GameLift Servers-linguagem suportada.

Para executar o Local, você também deve ter instalado os itens a seguir.

- Uma compilação do SDK do servidor para Amazon GameLift Servers versão 3.1.5 a 4.x.
- Java 8

Testar um servidor de jogos

Se você quiser testar somente seu servidor de jogos, você pode usar o AWS CLI para simular chamadas do cliente do jogo para o Amazon GameLift Servers Serviço local. Isso verifica se o servidor de jogos está funcionando conforme esperado com o seguinte:

- O servidor do jogo é iniciado corretamente e inicializa o SDK do servidor para Amazon GameLift Servers.
- Como parte do processo de lançamento, o servidor do jogo notifica Amazon GameLift Servers que o servidor está pronto para hospedar sessões de jogo.
- O servidor do jogo envia o status de saúde para Amazon GameLift Servers a cada minuto enquanto corre.

- O servidor de jogos responde a solicitações para iniciar uma nova sessão de jogo.

1. Iniciar Amazon GameLift Servers Local.

Abra uma janela de prompt de comando, navegue até o diretório que contém o arquivo *GameLiftLocal.jar* e o execute. Por padrão, o Local escuta solicitações de clientes de jogos na porta 8080. Para especificar um número de porta diferente, use o parâmetro `-p`, conforme mostrado no seguinte exemplo:

```
java -jar GameLiftLocal.jar -p 9080
```

Após a inicialização do Local, você vê logs indicando que dois servidores locais foram iniciados, um escutando o servidor de jogos e um escutando o cliente do jogo ou a AWS CLI. Os logs continuam informando a atividade nos dois servidores locais, inclusive a comunicação com e de componentes do jogo.

2. Inicie o servidor de jogos.

Comece seu Amazon GameLift Servers-servidor de jogos integrado localmente. Você não precisa alterar o endpoint do servidor de jogos.

Na janela do prompt de comando local, as mensagens de registro indicam que seu servidor de jogo se conectou ao Amazon GameLift Servers Serviço local. Isso significa que seu servidor de jogo inicializou com sucesso o SDK do servidor para Amazon GameLift Servers (`comInitSDK()`). Ele chamou `ProcessReady()` com os caminhos de log mostrados e, se for bem-sucedido, estará pronto para hospedar uma sessão de jogo. Enquanto o servidor do jogo está em execução, Amazon GameLift Servers registra cada relatório de estado de saúde do servidor do jogo. O seguinte exemplo do sistema de mensagens de log mostra um servidor de jogos integrado com êxito:

```
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
```

Entre as mensagens de erro e aviso em potencial estão as seguintes:

- Erro: "ProcessReady não encontrou um processo com PiD:<process ID>! O InitSDK() foi invocado?"
- Aviso: "O estado do processo já existe para o processo com PiD:<process ID>! É ProcessReady (...) invocado mais de uma vez?"

3. Inicie AWS CLI o.

Assim que o servidor de jogos chamar `ProcessReady()` com êxito, você poderá começar a fazer chamadas de cliente. Abra outra janela do prompt de comando e inicie a ferramenta AWS CLI . O, AWS CLI por padrão, usa o Amazon GameLift Servers ponto final de serviço web. Você deve substituí-lo pelo endpoint do Local em todas as solicitações usando o parâmetro `--endpoint-url`, conforme mostrado na solicitação de exemplo a seguir.

```
AWS gamelift describe-game-sessions --endpoint-url http://localhost:9080 --fleet-id fleet-123
```

Na janela do prompt de AWS CLI comando, `AWS gamelift` os comandos resultam em respostas conforme documentado na [Referência de AWS CLI Comandos](#).

4. Crie uma sessão de jogo.

Com o AWS CLI, envie uma solicitação [CreateGameSession\(\)](#). A solicitação deve seguir a sintaxe esperada. Para Local, o parâmetro `FleetId` pode ser definido como qualquer string válida (`^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
```

Na janela do prompt de comando local, as mensagens de log indicam que Amazon GameLift Servers O Local enviou um `onStartGameSession` retorno de chamada para seu servidor de jogo. Se uma sessão de jogo for criada com êxito, o servidor de jogos responderá invocando `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"
```

Na AWS CLI janela, Amazon GameLift Servers responde com um objeto de sessão de jogo, incluindo um ID de sessão de jogo. O novo status da sessão do jogo é Ativação. O status muda para Ativo quando o servidor do jogo é `ActivateGameSession` invocado. Se você quiser ver o status alterado, use o AWS CLI para `ligarDescribeGameSessions()`.

```
{
  "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
  }
}
```

```
}
```

Testar um servidor e um cliente de jogo

Para verificar a integração completa do jogo, inclusive a conexão de jogadores com jogos, você pode executar o servidor de jogos e o cliente localmente. Isso permite que você teste chamadas programáticas do seu cliente de jogo para o Amazon GameLift Servers Local. Você pode verificar as seguintes ações:

- O cliente do jogo está fazendo solicitações de AWS SDK com sucesso para o Amazon GameLift Servers Serviço local, inclusive para criar sessões de jogo, recuperar informações sobre sessões de jogo existentes e criar sessões de jogadores.
- O servidor de jogos está validando corretamente jogadores quando eles tentam participar de uma sessão do jogo. Para jogadores validados, o servidor de jogos pode recuperar dados do jogador (se implementado).
- O servidor de jogos informa uma conexão perdida quando um jogador deixa o jogo.
- O servidor de jogos informa o encerramento de uma sessão do jogo.

1. Iniciar Amazon GameLift Servers Local.

Abra uma janela de prompt de comando, navegue até o diretório que contém o arquivo *GameLiftLocal.jar* e o execute. Por padrão, o Local escuta solicitações de clientes de jogos na porta 8080. Para especificar um número de porta diferente, use o parâmetro `-p`, conforme mostrado no seguinte exemplo.

```
./gamelift-local -p 9080
```

Após a inicialização do Local, você vê logs mostrando que dois servidores locais foram iniciados, um escutando o servidor de jogos e um escutando o cliente do jogo ou a AWS CLI.

2. Inicie o servidor de jogos.

Comece seu Amazon GameLift Servers-servidor de jogos integrado localmente. Consulte [Testar um servidor de jogos](#) para obter mais detalhes sobre logs de mensagens.

3. Configure o cliente do jogo do Local e o inicie.

Para usar seu cliente de jogo com o Amazon GameLift Servers No serviço local, você deve fazer as seguintes alterações na configuração do seu cliente de jogo, conforme descrito em [Configurar Amazon GameLift Servers em um serviço de back-end](#):

- Altere o objeto `ClientConfiguration` a fim de apontar para o endpoint do Local, como `http://localhost:9080`.
- Defina um valor de ID da frota de destino. Para Local, você não precisa de um ID de frota real; defina a frota de destino como qualquer string válida (`^fleet-\S+`), como `fleet-1a2b3c4d-5e6f-7a8b-9c0d-1e2f3a4b5c6d`.
- Defina AWS as credenciais. Para Local, você não precisa de credenciais da AWS reais; você pode definir a chave de acesso e a chave secreta como qualquer string.

Na janela do prompt de comando local, depois de iniciar o cliente do jogo, as mensagens de registro devem indicar que ele inicializou o `GameLiftClient` e foi comunicado com sucesso com o Amazon GameLift Servers serviço.

4. Teste as chamadas do cliente do jogo para o Amazon GameLift Servers serviço.

Verifique se o cliente do jogo foi bem-sucedido fazendo qualquer uma ou todas as seguintes chamadas à API:

- [CreateGameSession\(\)](#)
- [DescribeGameSessions\(\)](#)
- [CreatePlayerSession\(\)](#)
- [CreatePlayerSessions\(\)](#)
- [DescribePlayerSessions\(\)](#)

Na janela do prompt de comando do Local, somente chamadas para `CreateGameSession()` resultam em mensagens de log. As mensagens de registro mostram quando Amazon GameLift Servers Local solicita que seu servidor de jogo inicie uma sessão de jogo (`onStartGameSession` retorno de chamada) e obtém sucesso `ActivateGameSession` quando seu servidor de jogo a invoca. Na janela da AWS CLI, todas as chamadas à API resultam em respostas ou mensagens de erro conforme documentado.

5. Verifique se o servidor de jogos está validando novas conexões de jogador.

Depois de criar uma sessão de jogo e uma sessão de jogador, estabeleça uma conexão direta com a sessão do jogo.

Na janela do prompt de comando do Local, as mensagens de log devem mostrar que o servidor de jogos enviou uma solicitação `AcceptPlayerSession()` para validar a nova conexão do jogador. Se você usar a chamada AWS CLI `DescribePlayerSessions()`, o status da sessão do jogador deverá mudar de Reservado para Ativo.

6. Verifique se o servidor do jogo está reportando o status do jogo e do jogador ao Amazon GameLift Servers serviço.

Para Amazon GameLift Servers para gerenciar a demanda dos jogadores e reportar corretamente as métricas, seu servidor de jogo deve reportar vários status para Amazon GameLift Servers. Verifique se o Local está registrando eventos relacionados às ações a seguir. Você também pode usar o AWS CLI para rastrear as alterações de status.

- O jogador se desconecta de uma sessão de jogo — Amazon GameLift Servers
As mensagens de registro locais devem mostrar que o servidor do jogo está ligando `RemovePlayerSession()`. Uma chamada à AWS CLI para `DescribePlayerSessions()` deve refletir uma alteração feita no status de `Active` para `Completed`. Também convém chamar `DescribeGameSessions()` para verificar se a contagem de jogadores atual da sessão do jogo diminui em um.
- A sessão do jogo termina — Amazon GameLift Servers
As mensagens de registro locais devem mostrar que o servidor do jogo está ligando `TerminateGameSession()`.

 Note

A orientação anterior era chamar `TerminateGameSession()` ao encerrar uma sessão de jogo. Esse método está obsoleto com Amazon GameLift Servers SDK do servidor v4.0.1. Consulte [Encerrar uma sessão de jogo](#).

- O processo do servidor foi encerrado — Amazon GameLift Servers
As mensagens de registro locais devem mostrar que o servidor do jogo está ligando `ProcessEnding()`. Uma AWS CLI chamada para `DescribeGameSessions()` deve refletir uma mudança de status de `Active` para `Terminated` (ou `Terminating`).

Variações com o local

Ao usar Amazon GameLift Servers Local, lembre-se do seguinte:

- Ao contrário do Amazon GameLift Servers serviço web, o Local não rastreia o status de saúde de um servidor e inicia o `onProcessTerminate` retorno de chamada. O Local simplesmente deixa de registrar em log relatórios de integridade do servidor de jogos.
- Para chamadas para o AWS SDK, a frota não IDs é validada e pode ser qualquer valor de string que atenda aos requisitos do parâmetro `()^fleet-\S+`.
- As sessões de jogo IDs criadas com o Local têm uma estrutura diferente. Eles incluem a string `local`, conforme mostrado aqui:

```
arn:aws:gamelift:local::gamesession/fleet-123/gsess-56961f8e-  
db9c-4173-97e7-270b82f0daa6
```

Adicionando FlexMatch matchmaking

Use Amazon GameLift Servers FlexMatch para adicionar a funcionalidade de matchmaking de jogadores aos seus jogos Amazon GameLift Servers hospedados. Você pode usar FlexMatch com servidores de jogos personalizados ou Amazon GameLift ServersRealtime.

O FlexMatch compara o serviço de marcação de jogos com um mecanismo de regras personalizável. Você cria como combinar jogadores com base nos atributos dos jogadores e nos modos de jogo que fazem sentido para o seu jogo. FlexMatchgerencia os detalhes básicos da avaliação de jogadores que estão procurando um jogo, formando partidas com uma ou mais equipes e iniciando sessões de jogo para sediar as partidas.

Para usar o FlexMatch serviço completo, você deve ter seus recursos de hospedagem configurados com filas. Amazon GameLift Serversusa filas para localizar os melhores locais de hospedagem possíveis para jogos em várias regiões e tipos de computação. Em particular, Amazon GameLift Servers as filas podem usar dados de latência, quando fornecidos pelos clientes do jogo, para colocar sessões de jogo de forma que os jogadores tenham a menor latência possível ao jogar.

Para obter mais informações sobre FlexMatch como incluir ajuda detalhada na integração de matchmaking em seus jogos, consulte estes tópicos do [Guia do Amazon GameLift ServersFlexMatch Desenvolvedor](#):

- [Como Amazon GameLift ServersFlexMatch funciona](#)

- [FlexMatchetapas de integração](#)

Obtenha dados da frota para uma Amazon GameLift Servers instância

Há algumas situações em que a criação ou o Amazon GameLift Servers Realtime script de seu jogo personalizado podem exigir informações sobre a Amazon GameLift Servers frota. Por exemplo, a compilação ou o script do seu jogo podem incluir código para:

- Monitore a atividade com base nos dados da frota.
- Acumule métricas para rastrear a atividade por dados da frota. (Muitos jogos usam esses dados para LiveOps atividades.)
- Forneça dados relevantes para serviços de jogos personalizados, como marcação de jogos, escalabilidade de capacidade adicional ou testes.

As informações da frota estão disponíveis como um arquivo JSON em cada instância nos seguintes locais:

- Windows: C:\GameMetadata\gamelift-metadata.json
- Linux: /local/gamemetadata/gamelift-metadata.json

O gamelift-metadata.json arquivo inclui os [atributos de um recurso Amazon GameLift Servers da frota](#).

Exemplo de arquivo JSON:

```
{
  "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"  
}
```

Integrando jogos com Amazon GameLift ServersRealtime

Este tópico fornece uma visão geral da Amazon GameLift Servers Realtime solução gerenciada Amazon GameLift Servers com. A visão geral explica quando essa solução é adequada para seu jogo e como é Amazon GameLift Servers Realtime compatível com jogos multijogador.



Tip

[Comece agora com Amazon GameLift Servers](#)

O que são Realtime servidores?

Realtimeos servidores são servidores de ready-to-go jogos leves Amazon GameLift Servers que permitem que você use com seus jogos multijogador. Realtimeos servidores removem o processo de desenvolvimento, teste e implantação de um servidor de jogos personalizado. Essa solução pode ajudar a minimizar o tempo e o esforço necessários para concluir o jogo.

Principais atributos

- Pilha de rede completa para interação entre servidores e clientes de jogos
- Funcionalidade principal do servidor de jogos
- Lógica de servidor personalizável
- Atualizações ao vivo Realtime das configurações e da lógica do servidor
- FlexMatchmatchmaking
- Controle flexível de recursos de hospedagem

Configure Realtime servidores criando uma frota e fornecendo um script de configuração.

Como Amazon GameLift ServersRealtime gerencia as sessões de jogo

Você pode adicionar lógica personalizada para o gerenciamento de sessões de jogos incorporando-a ao Realtime script. Você pode gravar o código para acessar objetos específicos do servidor,

adicionar lógica orientada por eventos usando retornos de chamada ou adicionar lógica com base em cenários não eventos.

Como Realtime clientes e servidores interagem

Durante uma sessão de jogo, os clientes do jogo interagem enviando mensagens ao Realtime servidor por meio de um serviço de back-end. Em seguida, o serviço de back-end retransmite as mensagens entre os clientes de jogos para trocar atividade, estado do jogo e dados relevantes do jogo.

Além disso, você pode personalizar como os clientes e servidores interagem adicionando lógica de jogo ao script do Realtime. Com a lógica de jogo personalizada, um Realtime servidor pode implementar retornos de chamada para iniciar respostas orientadas por eventos.

Protocolo de comunicação

Realtime servidores e clientes de jogos conectados se comunicam por meio de dois canais: uma conexão TCP para entrega confiável e um canal UDP para entrega rápida. Ao criar mensagens, os clientes de jogos escolhem qual protocolo usar de acordo com a natureza da mensagem. A entrega da mensagem é definida como UDP por padrão. Se um canal UDP não estiver disponível, Amazon GameLift Servers envia mensagens usando TCP como alternativa.

Conteúdo da mensagem

O conteúdo da mensagem consiste em dois elementos: um código de operação necessário (opCode) e uma carga opcional. O opCode de uma mensagem identifica um determinado evento de jogo ou atividade do jogador e a carga fornece dados adicionais relacionados ao código da operação. Ambos os elementos são definidos pelo desenvolvedor. O cliente do jogo age com base nos opCodes das mensagens que recebe.

Grupos de jogadores

Amazon GameLift Servers Realtime fornece funcionalidade para gerenciar grupos de jogadores. Por padrão, Amazon GameLift Servers coloca todos os jogadores que se conectam a um jogo em um grupo “todos os jogadores”. Além disso, os desenvolvedores podem configurar outros grupos para seus jogos, e os jogadores podem ser membros de vários grupos simultaneamente. Os membros do grupo podem enviar mensagens e compartilhar dados de jogos com todos os jogadores do grupo. Um uso possível para grupos é configurar equipes de jogadores e gerenciar a comunicação da equipe.

Amazon GameLift ServersRealtimecom certificados TLS

Com Amazon GameLift ServersRealtime, a autenticação do servidor e a criptografia de pacotes de dados são incorporadas ao serviço. Esses atributos de segurança são ativados quando você ativa a geração do certificado TLS. Quando um cliente de jogo tenta se conectar a um Realtime servidor, o servidor responde automaticamente com o certificado TLS, que o cliente valida. Amazon GameLift Serversmanipula a criptografia usando TLS para comunicação TCP (WebSockets) e DTLS para tráfego UDP.

Personalizando um servidor Realtime

Um Realtime servidor funciona como um servidor de retransmissão sem estado. O Realtime servidor retransmite pacotes de mensagens e dados do jogo entre os clientes do jogo conectados ao jogo. No entanto, o Realtime servidor não avalia mensagens, processa dados nem executa qualquer lógica de jogo. Usado dessa forma, cada cliente do jogo mantém sua própria visualização do estado do jogo e fornece atualizações para outros jogadores por meio do servidor de retransmissão. Cada cliente é responsável por incorporar essas atualizações e conciliar seu próprio estado do jogo.

Você pode personalizar seus servidores adicionando à funcionalidade de Realtime script. Com a lógica do jogo, por exemplo, é possível optar por criar um jogo com estado com uma visualização autoritativa do servidor do estado do jogo.

Amazon GameLift Serversdefine um conjunto de retornos de chamada do lado do servidor para scripts. Realtime Implemente esses retornos de chamada para adicionar funcionalidade baseada em eventos a seu servidor. Por exemplo, é possível:

- Autenticar um jogador quando um cliente do jogo tentar se conectar ao servidor.
- Validar se um jogador pode ingressar em um grupo mediante solicitação.
- Determinar quando entregar mensagens de um determinado jogador ou para um jogador de destino ou executar processamento adicional em resposta.
- Notificar todos os jogadores quando um jogador deixa um grupo ou se desconecta do servidor.
- Visualizar o conteúdo dos objetos da sessão de jogo ou objetos de mensagem, e usar os dados.

Implantação e atualização Amazon GameLift ServersRealtime

Uma das principais vantagens do Amazon GameLift Servers Realtime é a capacidade de atualizar seus scripts a qualquer momento. Quando você atualiza um script, Amazon GameLift Servers distribui a nova versão para todos os recursos de hospedagem em minutos. Depois de Amazon

GameLift Servers implantar o novo script, todas as novas sessões de jogo criadas após esse ponto usarão a nova versão do script. (As sessões de jogo existentes continuarão usando a versão original.)

Comece a integrar seu jogo com Amazon GameLift ServersRealtime:

- [Integrando um cliente de jogo para Amazon GameLift ServersRealtime](#)
- [Criação de um Realtime script](#)

Integrando um cliente de jogo para Amazon GameLift ServersRealtime

Este tópico descreve como preparar o cliente do jogo para poder ingressar e participar de sessões de jogos hospedadas pelo Amazon GameLift Servers.

Há dois conjuntos de tarefas necessárias para preparar o cliente do jogo:

- Configure o cliente do jogo para que ele adquira informações sobre jogos existentes, solicite criação de jogos, inicie novas sessões de jogo e reserve slots de sessões de jogo para um jogador.
- Permita que o cliente do jogo participe de uma sessão do jogo hospedada em um servidor Realtime e troque mensagens.

Encontrar ou criar sessões de jogo e sessões de jogadores

Configure o cliente do jogo para encontrar ou iniciar sessões de jogos, solicitar a marcação de jogos do FlexMatch e reservar espaço para jogadores em um jogo com a criação de sessões de jogador. Como prática recomendada, crie um serviço de back-end e use-o para fazer solicitações diretas ao Amazon GameLift Servers serviço quando acionado por uma ação do cliente do jogo. O serviço de back-end transmite as respostas relevantes de volta para o cliente do jogo.

1. Adicione o AWS SDK ao seu cliente de jogo, inicialize um Amazon GameLift Servers cliente e configure-o para usar os recursos de hospedagem em suas frotas e filas. O AWS SDK está disponível em vários idiomas; veja o. Amazon GameLift Servers SDKs [Para serviços de clientes de jogos](#)
2. Adicione Amazon GameLift Servers funcionalidade ao seu serviço de back-end. Para instruções mais detalhadas, consulte [Adicione Amazon GameLift Servers ao seu cliente de jogo](#) [Adicionar FlexMatch matchmaking](#). A melhor prática é o uso dos posicionamentos de sessões de jogo para criar novas sessões de jogo. Esse método permite que você aproveite ao máximo a

Amazon GameLift Servers capacidade de colocar novas sessões de jogo de forma rápida e inteligente, bem como usar os dados de latência do jogador para minimizar o atraso no jogo. No mínimo, o serviço de back-end deve poder solicitar novas sessões de jogo e processar dados da sessão de jogo em resposta. Você também pode adicionar funcionalidade para pesquisar e obter informações sobre sessões de jogo existentes e solicitar sessões de jogadores, o que efetivamente reserva um slot de jogador em uma sessão de jogo existente.

3. Transmita informações sobre a conexão de volta para o cliente do jogo. O serviço de back-end recebe objetos da sessão do jogo e da sessão do jogador em resposta às solicitações ao Amazon GameLift Servers serviço. Esses objetos contêm informações, especialmente os detalhes da conexão (endereço IP e porta) e o ID da sessão de jogadores, que o cliente do jogo precisa para se conectar à sessão de jogo em execução em um servidor em tempo real.

Conecte-se aos jogos em Amazon GameLift ServersRealtime

Permita que o cliente do jogo se conecte diretamente a uma sessão de jogo hospedada em um servidor Realtime e troque mensagens com o servidor e com outros jogadores.

1. Obtenha o SDK do cliente Amazon GameLift ServersRealtime, crie-o e adicione-o ao seu projeto de cliente de jogo. Consulte o arquivo README para obter mais informações sobre requisitos de SDK e instruções sobre como criar as bibliotecas de cliente.
2. Ligue [Client\(\)](#) com uma configuração de cliente que especifica o tipo de client/server conexão a ser usada.

Note

Se estiver se conectando a um servidor em tempo real em execução em uma frota protegida por um certificado TLS, você deverá especificar um tipo de conexão segura.

3. Adicione a seguinte funcionalidade ao cliente do jogo. Consulte [Amazon GameLift ServersRealtime referância da API do cliente \(C#\)](#) para obter mais informações.
 - Conectar-se e desconectar-se de um jogo
 - [Connect\(\)](#)
 - [Disconnect\(\)](#)
 - Enviar mensagens a destinatários de destino
 - [SendMessage\(\)](#)
 - Receber e processar mensagens

- [OnDataReceived\(\)](#)
 - Ingressar e sair de grupos de jogadores
 - [JoinGroup\(\)](#)
 - [RequestGroupMembership\(\)](#)
 - [LeaveGroup\(\)](#)
4. Configure manipuladores de eventos para retornos de chamada do cliente conforme necessário. Consulte [Amazon GameLift Servers Realtime referência da API do cliente \(C#\): retornos de chamada assíncronos](#).

Ao trabalhar com frotas em tempo real com a geração de certificados TLS habilitada, o servidor é autenticado automaticamente usando o certificado TLS. O tráfego TCP e UDP é criptografado em trânsito para fornecer segurança na camada de transporte. O tráfego TCP é criptografado usando TLS 1.2 e o tráfego UDP é criptografado usando DTLS 1.2.

Exemplos de cliente do jogo

Cliente em tempo real básico (C#)

Este exemplo ilustra uma integração básica do cliente de jogo com o SDK do cliente (C#) para Amazon GameLift Servers Realtime. Como mostrado, o exemplo inicializa um objeto de cliente do Realtime, configura manipuladores de eventos e implementa o retorno de chamada do lado do cliente, se conecta a um servidor Realtime, envia uma mensagem e se desconecta.

```
using System;
using System.Text;
using Aws.GameLift.Realtime;
using Aws.GameLift.Realtime.Event;
using Aws.GameLift.Realtime.Types;

namespace Example
{
    /**
     * An example client that wraps the client SDK for Amazon GameLift Servers Realtime
     *
     * You can redirect logging from the SDK by setting up the LogHandler as such:
     * ClientLogger.LogHandler = (x) => Console.WriteLine(x);
     *
     */
    class RealTimeClient
```

```

{
    public Aws.GameLift.Realtime.Client Client { get; private set; }

    // An opcode defined by client and your server script that represents a custom
message type
    private const int MY_TEST_OP_CODE = 10;

    /// Initialize a client for Amazon GameLift Servers Realtime and connect to a
player session.
    /// <param name="endpoint">The DNS name that is assigned to Realtime server</
param>
    /// <param name="remoteTcpPort">A TCP port for the Realtime server</param>
    /// <param name="listeningUdpPort">A local port for listening to UDP traffic</
param>
    /// <param name="connectionType">Type of connection to establish between client
and the Realtime server</param>
    /// <param name="playerSessionId">The player session ID that is assigned to the
game client for a game session </param>
    /// <param name="connectionPayload">Developer-defined data to be used during
client connection, such as for player authentication</param>
    public RealTimeClient(string endpoint, int remoteTcpPort, int listeningUdpPort,
ConnectionType connectionType,
        string playerSessionId, byte[] connectionPayload)
    {
        // Create a client configuration to specify a secure or unsecure connection
type
        // Best practice is to set up a secure connection using the connection type
RT_OVER_WSS_DTLS_TLS12.
        ClientConfiguration clientConfiguration = new ClientConfiguration()
        {
            // C# notation to set the field ConnectionType in the new instance of
ClientConfiguration
            ConnectionType = connectionType
        };

        // Create a Realtime client with the client configuration
        Client = new Client(clientConfiguration);

        // Initialize event handlers for the Realtime client
        Client.ConnectionOpen += OnOpenEvent;
        Client.ConnectionClose += OnCloseEvent;
        Client.GroupMembershipUpdated += OnGroupMembershipUpdate;
        Client.DataReceived += OnDataReceived;
    }
}

```

```

        // Create a connection token to authenticate the client with the Realtime
server
        // Player session IDs can be retrieved using AWS SDK for Amazon GameLift
Servers
        ConnectionToken connectionToken = new ConnectionToken(playerSessionId,
connectionPayload);

        // Initiate a connection with the Realtime server with the given connection
information
        Client.Connect(endpoint, remoteTcpPort, listeningUdpPort, connectionToken);
    }

    public void Disconnect()
    {
        if (Client.Connected)
        {
            Client.Disconnect();
        }
    }

    public bool IsConnected()
    {
        return Client.Connected;
    }

    /// <summary>
    /// Example of sending to a custom message to the server.
    ///
    /// Server could be replaced by known peer Id etc.
    /// </summary>
    /// <param name="intent">Choice of delivery intent i.e. Reliable, Fast etc. </
param>
    /// <param name="payload">Custom payload to send with message</param>
    public void SendMessage(DeliveryIntent intent, string payload)
    {
        Client.SendMessage(Client.NewMessage(MY_TEST_OP_CODE)
            .WithDeliveryIntent(intent)
            .WithTargetPlayer(Constants.PLAYER_ID_SERVER)
            .WithPayload(StringToBytes(payload)));
    }

    /**
     * Handle connection open events
     */

```

```
public void OnOpenEvent(object sender, EventArgs e)
{
}

/**
 * Handle connection close events
 */
public void OnCloseEvent(object sender, EventArgs e)
{
}

/**
 * Handle Group membership update events
 */
public void OnGroupMembershipUpdate(object sender, GroupMembershipEventArgs e)
{
}

/**
 * Handle data received from the Realtime server
 */
public virtual void OnDataReceived(object sender, DataReceivedEventArgs e)
{
    switch (e.OpCode)
    {
        {
            // handle message based on OpCode
            default:
                break;
        }
    }
}

/**
 * Helper method to simplify task of sending/receiving payloads.
 */
public static byte[] StringToBytes(string str)
{
    return Encoding.UTF8.GetBytes(str);
}

/**
 * Helper method to simplify task of sending/receiving payloads.
 */
public static string BytesToString(byte[] bytes)
{
}
```



```
        return Encoding.UTF8.GetString(bytes);
    }
}
```

Criação de um Realtime script

Para usar em seu jogo, você precisa fornecer um script (na forma de algum JavaScript código) para configurar e, opcionalmente, personalizar uma frota de Amazon GameLift Servers Realtime. Este tópico abrange as principais etapas na criação de um script do Realtime. Assim que o script estiver pronto, faça upload dele no serviço Amazon GameLift Servers e use-o para criar uma frota (consulte [Implante um script para Amazon GameLift Servers Realtime](#)).

Para preparar um script para uso com Amazon GameLift Servers Realtime ele, adicione a seguinte funcionalidade ao seu Realtime script.

Gerenciar o ciclo de vida da sessão de jogos (obrigatório)

No mínimo, um script Realtime deve incluir a função `Init()`, que prepara o servidor Realtime para iniciar uma sessão de jogo. Também é altamente recomendável que você também forneça uma maneira de encerrar sessões de jogos, para garantir que novas sessões de jogos possam continuar a ser iniciadas na frota.

A função de retorno de chamada `Init()`, quando chamada, recebe um objeto de sessão Realtime, que contém uma interface para o servidor Realtime. Consulte [Amazon GameLift Servers Realtime interface](#) para obter mais detalhes sobre essa interface.

Para encerrar corretamente a uma sessão de jogo, o script também deve chamar a função `session.processEnding` do servidor Realtime. Isso requer um mecanismo para determinar quando encerrar uma sessão. O código do script de exemplo ilustra um mecanismo simples que verifica se há conexões de jogador e aciona o encerramento da sessão de jogo quando nenhum jogador tiver se conectado à sessão por um período especificado.

Amazon GameLift Servers Realtime com a configuração mais básica — inicialização e encerramento do processo do servidor — atuam essencialmente como servidores de retransmissão sem estado. O servidor Realtime retransmite mensagens e dados de jogos entre os clientes de jogos que estão conectados ao jogo, mas não realiza ação independente para processar dados ou executar lógica. Opcionalmente, é possível adicionar a lógica do jogo, acionada por eventos de jogos ou outros mecanismos, conforme necessário para o jogo.

Adicionar lógica do jogo do lado do servidor (opcional)

Opcionalmente, você pode adicionar lógica do jogo ao script Realtime. Por exemplo, você pode fazer qualquer um ou todos os seguintes. O código de script de exemplo fornece ilustração. Consulte [Referência de script para Amazon GameLift ServersRealtime](#).

- Adicionar lógica orientada por eventos. Implemente as funções de retorno de chamada para responder a eventos de cliente-servidor. Consulte [Retornos de chamada de script para Amazon GameLift ServersRealtime](#) para obter uma lista completa de retornos de chamada.
- Acione lógica enviando mensagens ao servidor. Crie um conjunto de códigos de operação especiais para mensagens enviadas de clientes de jogo para o servidor, e adicione funções para manipular o recebimento. Use o retorno de chamada `onMessage` e analise o conteúdo da mensagem usando a interface `gameMessage` (consulte [gameMessage.opcode](#)).
- Ative a lógica do jogo para acessar seus outros AWS recursos. Para obter detalhes, consulte [Comunique-se com outros AWS recursos de suas frotas](#).
- Permita que a lógica do jogo acesse as informações da frota da instância em que está sendo executada. Para obter detalhes, consulte [Obtenha dados da frota para uma Amazon GameLift Servers instância](#).

Amazon GameLift ServersRealtimeexemplo de script

Este exemplo ilustra um script básico necessário para a implantação, Amazon GameLift Servers Realtime além de uma lógica personalizada. Ele contém a função `Init()` necessária, e usa um mecanismo de temporizador para acionar o encerramento da sessão de jogo com base no tempo sem conexões de jogador. Ele também inclui alguns ganchos de lógica personalizada, incluindo algumas implementações de chamada de retorno.

```
// Example Realtime Server Script
'use strict';

// Example override configuration
const configuration = {
  pingIntervalTime: 30000,
  maxPlayers: 32
};

// Timing mechanism used to trigger end of game session. Defines how long, in
// milliseconds, between each tick in the example tick loop
const tickTime = 1000;
```

```
// Defines how long to wait in Seconds before beginning early termination check in
// the example tick loop
const minimumElapsedTime = 120;

var session; // The Realtime server session object
var logger; // Log at appropriate level
    via .info(), .warn(), .error(), .debug()
var startTime; // Records the time the process started
var activePlayers = 0; // Records the number of connected players
var onProcessStartedCalled = false; // Record if onProcessStarted has been called

// Example custom op codes for user-defined messages
// Any positive op code number can be defined here. These should match your client
// code.
const OP_CODE_CUSTOM_OP1 = 111;
const OP_CODE_CUSTOM_OP1_REPLY = 112;
const OP_CODE_PLAYER_ACCEPTED = 113;
const OP_CODE_DISCONNECT_NOTIFICATION = 114;

// Example groups for user-defined groups
// Any positive group number can be defined here. These should match your client code.
// When referring to user-defined groups, "-1" represents all groups, "0" is reserved.
const RED_TEAM_GROUP = 1;
const BLUE_TEAM_GROUP = 2;

// Called when game server is initialized, passed server's object of current session
function init(rtSession) {
    session = rtSession;
    logger = session.getLogger();
}

// On Process Started is called when the process has begun and we need to perform any
// bootstrapping. This is where the developer should insert any code to prepare
// the process to be able to host a game session, for example load some settings or set
// state
//
// Return true if the process has been appropriately prepared and it is okay to invoke
// the
// GameLift ProcessReady() call.
function onProcessStarted(args) {
    onProcessStartedCalled = true;
    logger.info("Starting process with args: " + args);
    logger.info("Ready to host games...");
```

```
    return true;
}

// Called when a new game session is started on the process
function onStartGameSession(gameSession) {
    // Complete any game session set-up

    // Set up an example tick loop to perform server initiated actions
    startTime = getTimeInS();
    tickLoop();
}

// Handle process termination if the process is being terminated by Amazon GameLift
Servers
// You do not need to call ProcessEnding here
function onProcessTerminate() {
    // Perform any clean up
}

// Return true if the process is healthy
function onHealthCheck() {
    return true;
}

// On Player Connect is called when a player has passed initial validation
// Return true if player should connect, false to reject
function onPlayerConnect(connectMsg) {
    // Perform any validation needed for connectMsg.payload, connectMsg.peerId
    return true;
}

// Called when a Player is accepted into the game
function onPlayerAccepted(player) {
    // This player was accepted -- let's send them a message
    const msg = session.newTextGameMessage(OP_CODE_PLAYER_ACCEPTED, player.peerId,
                                           "Peer " + player.peerId + " accepted");
    session.sendReliableMessage(msg, player.peerId);
    activePlayers++;
}

// On Player Disconnect is called when a player has left or been forcibly terminated
// Is only called for players that actually connected to the server and not those
rejected by validation
```

```
// This is called before the player is removed from the player list
function onPlayerDisconnect(peerId) {
    // send a message to each remaining player letting them know about the disconnect
    const outMessage = session.newTextGameMessage(OP_CODE_DISCONNECT_NOTIFICATION,
                                                    session.getServerId(),
                                                    "Peer " + peerId + " disconnected");
    session.getPlayers().forEach((player, playerId) => {
        if (playerId !== peerId) {
            session.sendReliableMessage(outMessage, playerId);
        }
    });
    activePlayers--;
}

// Handle a message to the server
function onMessage(gameMessage) {
    switch (gameMessage.opCode) {
        case OP_CODE_CUSTOM_OP1: {
            // do operation 1 with gameMessage.payload for example sendToGroup
            const outMessage = session.newTextGameMessage(OP_CODE_CUSTOM_OP1_REPLY,
session.getServerId(), gameMessage.payload);
            session.sendGroupMessage(outMessage, RED_TEAM_GROUP);
            break;
        }
    }
}

// Return true if the send should be allowed
function onSendToPlayer(gameMessage) {
    // This example rejects any payloads containing "Reject"
    return (!gameMessage.getPayloadAsText().includes("Reject"));
}

// Return true if the send to group should be allowed
// Use gameMessage.getPayloadAsText() to get the message contents
function onSendToGroup(gameMessage) {
    return true;
}

// Return true if the player is allowed to join the group
function onPlayerJoinGroup(groupId, peerId) {
    return true;
}
```

```
// Return true if the player is allowed to leave the group
function onPlayerLeaveGroup(groupId, peerId) {
    return true;
}

// A simple tick loop example
// Checks to see if a minimum amount of time has passed before seeing if the game has
// ended
async function tickLoop() {
    const elapsedTime = getTimeInS() - startTime;
    logger.info("Tick... " + elapsedTime + " activePlayers: " + activePlayers);

    // In Tick loop - see if all players have left early after a minimum period of time
    // has passed
    // Call processEnding() to terminate the process and quit
    if ( (activePlayers == 0) && (elapsedTime > minimumElapsedTime)) {
        logger.info("All players disconnected. Ending game");
        const outcome = await session.processEnding();
        logger.info("Completed process ending with: " + outcome);
        process.exit(0);
    }
    else {
        setTimeout(tickLoop, tickTime);
    }
}

// Calculates the current time in seconds
function getTimeInS() {
    return Math.round(new Date().getTime()/1000);
}

exports.ssExports = {
    configuration: configuration,
    init: init,
    onProcessStarted: onProcessStarted,
    onMessage: onMessage,
    onPlayerConnect: onPlayerConnect,
    onPlayerAccepted: onPlayerAccepted,
    onPlayerDisconnect: onPlayerDisconnect,
    onSendToPlayer: onSendToPlayer,
    onSendToGroup: onSendToGroup,
    onPlayerJoinGroup: onPlayerJoinGroup,
    onPlayerLeaveGroup: onPlayerLeaveGroup,
    onStartGameSession: onStartGameSession,
```

```
onProcessTerminate: onProcessTerminate,  
onHealthCheck: onHealthCheck  
};
```

Implantação de software de servidor de jogos para hospedagem Amazon GameLift Servers

Implante seu software de servidor de jogos multijogador instalando-o em seus recursos de hospedagem, iniciando processos do servidor de jogos e preparando-os para hospedar jogos para jogadores. As etapas para preparar o software do servidor de jogos para implantação dependem do tipo de Amazon GameLift Servers solução que você está usando. Em todos os cenários, o software de servidor de jogos implantado interage com o Amazon GameLift Servers serviço para lidar com o posicionamento da sessão de jogo e comunicar os detalhes da conexão de um cliente de jogo.

Seu software de servidor de jogos deve primeiro ser integrado Amazon GameLift Servers, conforme descrito em [Preparando jogos para Amazon GameLift Servers](#).

Os tópicos desta seção fornecem orientações sobre como preparar o software para implantação nos cenários a seguir.

- Software de servidor de jogos personalizado
 - Para EC2 hospedagem Amazon GameLift Servers gerenciada
 - Para hospedagem de contêineres Amazon GameLift Servers gerenciados
 - Para hospedagem Amazon GameLift Servers em qualquer lugar
- Amazon GameLift ServersRealtimescript personalizado para EC2 hospedagem Amazon GameLift Servers gerenciada

Tópicos

- [Implemente uma criação de servidor personalizada para Amazon GameLift Servers hospedagem](#)
- [Crie uma imagem de contêiner para Amazon GameLift Servers](#)
- [Implante um script para Amazon GameLift ServersRealtime](#)

Implemente uma criação de servidor personalizada para Amazon GameLift Servers hospedagem

Depois de integrar seu servidor de jogos com Amazon GameLift Servers (consulte [Preparando jogos para Amazon GameLift Servers](#)), instale o software do servidor de jogos em seus recursos

computacionais para hospedagem. Esse processo varia de acordo com o tipo de Amazon GameLift Servers hospedagem que você está usando.

Implantar para hospedagem gerenciada

Se você estiver usando Amazon GameLift Servers EC2 hospedagem gerenciada, você deve empacotar o software do servidor de jogos e enviá-lo para Amazon GameLift Servers. Quando você cria uma frota gerenciada, Amazon GameLift Servers o implanta automaticamente em cada instância da frota.

Os tópicos desta seção descrevem como empacotar seus arquivos de compilação para upload, criar um script de instalação de compilação opcional e, em seguida, fazer upload dos arquivos usando o [AWS Command Line Interface \(AWS CLI\)](#) ou o AWS SDK.

Implementar para hospedagem Anywhere

Se você estiver usando Amazon GameLift Servers Em qualquer frota de hospedagem autogerenciada, é sua responsabilidade instalar o software do servidor de jogos em cada computador de uma frota e mantê-lo atualizado.

Quando um processo integrado do servidor de jogos começa a ser executado, ele inicializa e estabelece comunicação automaticamente com o Amazon GameLift Servers serviço. O processo do servidor inicia as sessões de jogo a partir das instruções de Amazon GameLift Servers e reporta a atividade ao serviço.

Tópicos

- [Criar um pacote dos arquivos de compilação do jogo](#)
- [Adicionar um script de instalação da compilação](#)
- [Crie um Amazon GameLift Servers recurso de criação para hospedagem gerenciada](#)
- [Atualize uma compilação de servidor de jogo para Amazon GameLift Servers hospedagem gerenciada](#)

Criar um pacote dos arquivos de compilação do jogo

Antes de enviar seu servidor de jogo configurado para Amazon GameLift Servers, empacote os arquivos de compilação do jogo em um diretório de compilação. Esse processo é um requisito ao hospedar com uma frota EC2 gerenciada e é uma prática recomendada ao hospedar com uma frota

Anywhere. O diretório de compilação deve incluir todos os componentes necessários para executar os servidores de jogos e hospedar sessões de jogos. Isso pode incluir o seguinte:

- Binários do servidor de jogo – Os arquivos binários necessários para executar o servidor de jogo. Uma compilação pode incluir binários para vários servidores de jogos compilados para rodar na mesma plataforma. Para obter uma lista das plataformas com suporte, consulte [Obtenha ferramentas Amazon GameLift Servers de desenvolvimento](#).
- Dependências – Qualquer arquivo dependente exigido pelos executáveis do servidor de jogo para ser executado. Entre os exemplos estão ativos, arquivos de configuração e bibliotecas dependentes.

 Note

Para compilações de jogos criadas com o SDK do servidor para Amazon GameLift Servers para C++ (incluindo aqueles criados com o plug-in Unreal), inclua a DLL do OpenSSL para a mesma versão do OpenSSL com a qual você criou o SDK do servidor. Consulte o arquivo README em SDK em servidor para obter mais detalhes.

- Script de instalação (opcional) — Um arquivo de script para lidar com tarefas que instalam a versão do seu jogo Amazon GameLift Servers servidores de hospedagem. Coloque este arquivo na raiz do diretório de compilação. Amazon GameLift Servers executa o script de instalação como parte da criação da frota.

Você pode configurar qualquer aplicativo em sua compilação, incluindo seu script de instalação, para acessar seus recursos com segurança em outros AWS serviços. Para obter informações sobre formas de fazer isso, consulte [Comunique-se com outros AWS recursos de suas frotas](#).

Depois de empacotar os arquivos de compilação, certifique-se de que o servidor de jogos possa ser executado em uma instalação limpa do sistema operacional de destino para verificar se todas as dependências necessárias estão incluídas e se o script de instalação está correto.

Adicionar um script de instalação da compilação

Crie um script de instalação para o OS (sistema operacional) da compilação do seu jogo:

- Windows: crie um arquivo em lote chamado `install.bat`.
- Linux: crie um arquivo de script de shell chamado `install.sh`.

Ao criar um script de instalação, lembre-se do seguinte:

- O script não pode receber nenhuma entrada do usuário.
- Amazon GameLift Servers instala a compilação e recria os diretórios de arquivos em seu pacote de compilação em um servidor de hospedagem nos seguintes locais:
 - Frotas do Windows: C:\game
 - Frotas do Linux: /local/game
- Durante o processo de instalação de frotas do Linux, o usuário de execução tem acesso limitado à estrutura de arquivos da instância. Este usuário tem direitos totais para o diretório em que seus arquivos de compilação estão instalados. Se seu script de instalação realizar ações que exijam permissões de administrador, especifique o acesso de administrador usando sudo. O usuário de execução para frotas do Windows tem permissões de administrador por padrão. As falhas de permissão relacionadas ao script de instalação geram uma mensagem de evento que indica um problema com o script.
- No Linux, Amazon GameLift Servers suporta linguagens comuns de interpretação de shell, como bash. Adicione um shebang (como #!/bin/bash) à parte superior do seu script de instalação. Para verificar o suporte para os seus comandos shell preferidos, acesse remotamente uma instância ativa do Linux e abra um prompt de shell. Para obter mais informações, consulte [Conecte-se remotamente a Amazon GameLift Servers instâncias de frota](#).
- O script de instalação não pode depender de uma conexão de emparelhamento da VPC. Uma conexão de emparelhamento de VPC não estará disponível até depois Amazon GameLift Servers instala a compilação em instâncias de frota.

Exemplo Arquivo bash de instalação em Windows

Este exemplo de arquivo `install.bat` instala os componentes de tempo de execução do Visual C++ necessários para que o servidor do jogo e grave os resultados em um arquivo de log. O script inclui o arquivo do componente no pacote de compilação na raiz.

```
vcredis_x64.exe /install /quiet /norestart /log c:\game\vcredis_2013_x64.log
```

Exemplo Script de shell de instalação do Linux

Esse arquivo `install.sh` de exemplo usa bash no script de instalação e grava os resultados em um arquivo de log.

```
#!/bin/bash
```

```
echo 'Hello World' > install.log
```

Este `install.sh` arquivo de exemplo mostra como você pode usar o CloudWatch agente da Amazon para coletar métricas personalizadas e em nível de sistema e lidar com a rotação de registros. Porque Amazon GameLift Servers é executado em um serviço VPC, você deve conceder Amazon GameLift Servers permissões para assumir uma função AWS Identity and Access Management (IAM) em seu nome. Para permitir Amazon GameLift Servers para assumir uma função, crie uma função que inclua a política `AWS CloudWatchAgentAdminPolicy` gerenciada e use essa função ao criar uma frota.

```
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
```

Crie um Amazon GameLift Servers recurso de criação para hospedagem gerenciada

Ao criar uma compilação e fazer upload dos arquivos, você tem algumas opções:

- [Criar uma compilação usando um diretório de arquivos](#). Esse é a opção mais simples e o mais usada.
- [Crie uma compilação com arquivos no Amazon Simple Storage Service \(Amazon S3\)](#). Com essa opção, é possível gerenciar as versões de compilação no Amazon S3.

Com os dois métodos, Amazon GameLift Servers cria um novo recurso de compilação com um ID de compilação exclusivo e outros metadados. A compilação começa no status Inicializado. Depois Amazon GameLift Servers adquire os arquivos do servidor do jogo, a compilação passa para o status Pronto.

Quando a compilação estiver pronta, você poderá implantá-la em um novo Amazon GameLift Servers frota. Para obter mais informações, consulte [Crie uma EC2 frota Amazon GameLift Servers gerenciada](#). Quando Amazon GameLift Servers configura a nova frota, baixa os arquivos de compilação para cada instância da frota e instala os arquivos de compilação.

Criar uma compilação usando um diretório de arquivos

Para criar uma compilação de jogo armazenada em qualquer local, incluindo um diretório local, use o [upload-build](#) AWS CLI comando. Esse comando cria um novo registro de compilação no Amazon GameLift Servers e carrega arquivos de um local que você especificar.

Envie uma solicitação de upload. Em uma janela da linha de comando, insira o comando `upload-build` e os parâmetros a seguir.

```
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 \  
  --region region name
```

- `operating-system` – O ambiente de execução da construção do servidor de jogos. Especifique um valor de OS. Não é possível atualizá-lo mais tarde.
- `server-sdk-version`— A versão do Amazon GameLift Servers SDK do servidor ao qual seu servidor de jogos está integrado. Se você não fornecer um valor, Amazon GameLift Servers usa o valor padrão `4.0.2`. Se você especificar uma versão incorreta do SDK do servidor, a compilação do servidor do jogo poderá falhar ao ligar `InitSdk` para estabelecer uma conexão com o Amazon GameLift Servers serviço.
- `build-root` – O caminho do diretório dos arquivos de compilação.
- `name` – Um nome descritivo para a nova compilação.
- `build-version` – Os detalhes da versão dos arquivos de compilação.
- `region`— A AWS região onde você deseja criar sua construção. Crie a compilação na Região onde você planeja implantar frotas. Se você estiver implantando o jogo em várias Regiões, crie uma compilação em cada região.

Note

Visualize sua região padrão atual usando o [aws configure get region](#). Para alterar sua Região padrão, use o comando [aws configure set region *region name*](#).

Exemplos

```
aws gamelift upload-build \  
  --operating-system AMAZON_LINUX_2023 \  
  
  --server-sdk-version "5.0.0" \  
  --build-root "~/mygame" \  
  --name "MyGameBuild" \  
  --build-version "1.0" \  
  --region us-east-1
```

```
--name "My Game Nightly Build" \  
--build-version "build 255" \  
--region us-west-2
```

```
aws gamelift upload-build \  
  --operating-system WINDOWS_2016 \  
  --server-sdk-version "5.0.0" \  
  --build-root "C:\mygame" \  
  --name "My Game Nightly Build" \  
  --build-version "build 255" \  
  --region us-west-2
```

Em resposta à sua solicitação de upload, Amazon GameLift Servers fornece o progresso do upload. Em um upload bem-sucedido, Amazon GameLift Servers retorna o novo ID do registro de construção. O tempo de upload depende do tamanho dos arquivos de jogo e da velocidade da conexão.

Criar uma compilação com arquivos no Amazon S3

Você pode armazenar seus arquivos de compilação no Amazon S3 e enviá-los para Amazon GameLift Servers a partir daí. Ao criar sua compilação, você especifica a localização do bucket do S3 e Amazon GameLift Servers recupera os arquivos de compilação diretamente do Amazon S3.

Para criar um recurso de compilação

1. Armazene os arquivos de compilação no Amazon S3. Crie um arquivo.zip contendo os arquivos de compilação empacotados e carregue-o em um bucket do S3 no seu. Conta da AWS Anote o rótulo do bucket e o nome do arquivo. Você precisará deles ao criar um Amazon GameLift Servers construir.
2. Dar Amazon GameLift Servers acesso aos seus arquivos de compilação. Crie um perfil do IAM seguindo as instruções em [Acesse um arquivo de compilação do jogo no Amazon S3](#). Após ter criado a função, anote o nome do recurso da Amazon (ARN) da nova função, você precisará disso ao criar uma compilação.
3. Criar uma compilação. Use o comando Amazon GameLift Servers console ou o AWS CLI para criar um novo registro de construção. Você deve ter a PassRole permissão, conforme descrito em [Exemplos de permissão do IAM para Amazon GameLift Servers](#).

Console

1. No [Amazon GameLift Servers console](#), no painel de navegação, escolha Hosting, Builds.
2. Na página Compilações, escolha Criar compilação.
3. Na página Criar compilação, em Configurações de compilação, faça o seguinte:
 - a. Em Nome, insira um nome de script.
 - b. Em Versão, insira uma versão. Como você pode atualizar o conteúdo de uma compilação, os dados da versão podem ajudá-lo a monitorar as atualizações.
 - c. Em SO (Sistema operacional), escolha o sistema operacional da versão do seu servidor de jogos. Não é possível atualizar esse valor posteriormente.
 - d. Em Compilação de servidor de jogo, insira o URI do S3 do objeto de compilação que você carregou no Amazon S3 e escolha a versão do objeto. Caso não se lembre do URI e da versão do objeto do Amazon S3, escolha Browse S3 e pesquise o objeto de construção.
 - e. Para a função do IAM, escolha a função que você criou e que fornece Amazon GameLift Servers acesso ao seu bucket do S3 e ao objeto de construção.
4. (Opcional) Em Tags, adicione tags à compilação inserindo pares de Chave e Valor.
5. Escolha Criar.

Amazon GameLift Servers atribui uma ID à nova compilação e carrega o arquivo.zip designado. Visualize a nova compilação, incluindo o status, na página Compilações.

AWS CLI

Para definir a nova compilação e fazer upload dos arquivos de compilação do servidor, use o comando [create-build](#).

1. Abra uma janela de linha de comando e alterne para um diretório em que você possa usar a AWS CLI.
2. Digite o comando create-build:

```
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 \  
  --s3-uri S3 URI of build artifact \  
  --iam-role ARN of IAM role
```



```
--storage-location "Bucket"=S3 bucket label,"Key"=Build .zip file name,"RoleArn"=Access role ARN} \  
--region region name
```

- **name** – Um nome descritivo para a nova compilação
- **server-sdk-version**— A versão do SDK do servidor para Amazon GameLift Servers você costumava integrar seu servidor de jogos com Amazon GameLift Servers. Se você não fornecer um valor, Amazon GameLift Servers usa o valor padrão `4.0.2`.
- **operating-system** – O ambiente de runtime da compilação do servidor de jogo. Especifique um valor de OS. Não é possível atualizá-lo mais tarde.
- **build-version** – Os detalhes da versão dos arquivos de compilação. Essas informações podem ser úteis porque cada nova versão do seu servidor de jogos requer um novo recurso de compilação.
- **storage-location**
 - **Bucket** – O nome do bucket do S3 que contém a compilação. Por exemplo: `"my_build_files"`.
 - **Key** – O nome do arquivo .zip que contém os arquivos de compilação. Por exemplo, `"my_game_build_7.0.1, 7.0.2"`.
 - **RoleARN** – O ARN atribuído ao perfil do IAM criado. Por exemplo, `"arn:aws:iam: :111122223333:role/"`. `GameLiftAccess` Para visualizar um exemplo de política, consulte [Acesse um arquivo de compilação do jogo no Amazon S3](#).
- **region**— Crie a construção na AWS região em que você planeja implantar frotas. Se você estiver implantando o jogo em várias Regiões, crie uma compilação em cada região.

 Note

É recomendável verificar sua região padrão atual usando o [configure get](#) comando. Para alterar sua região padrão, use o comando [configure set](#).

Exemplo

```
aws gamelift create-build \  
--operating-system WINDOWS_2016 \  

```

```
--storage-location  
"Bucket"="my_game_build_files", "Key"="mygame_build_101.zip", "RoleArn"="arn:aws:iam::111  
gamelift" \  
--name "My Game Nightly Build" \  
--build-version "build 101" \  
--region us-west-2
```

3. Para visualizar a nova compilação, use o comando [describe-build](#).

Atualize uma compilação de servidor de jogo para Amazon GameLift Servers hospedagem gerenciada

Quando você implanta seu servidor de jogos, construa para Amazon GameLift Servers EC2 hospedagem gerenciada, você carrega o software do seu servidor de jogos e cria um Amazon GameLift Servers recurso de construção. Depois de criar um Amazon GameLift Servers build, você pode atualizar os metadados da compilação, mas não pode atualizar os arquivos de compilação em si. Para implantar atualizações em seu servidor de jogo, faça o upload dos arquivos atualizados e crie um novo Amazon GameLift Servers construa usando o AWS CLI comando de [upload-build](#) comando. Como alternativa, você pode usar o comando [create-build](#) para carregar uma nova compilação de um bucket do Amazon S3 que você controla. Em seguida, implante a nova compilação criando uma frota para ela.

Você pode atualizar os metadados de uma compilação, incluindo o nome e a descrição. Para essas tarefas, use o Amazon GameLift Servers console ou o [update-build](#) AWS CLI comando.

Automatizar as atualizações de compilação de jogos

Siga estas dicas para ajudar a automatizar e agilizar o processo de atualização de versões de servidores de jogos para Amazon GameLift Servers frotas gerenciadas:

- Use filas de sessões de jogo e troque de frotas conforme necessário. Ao enviar solicitações de sessão de jogo para Amazon GameLift Servers, especifique uma fila de sessões de jogo em vez de uma frota específica. Com as filas, é possível adicionar frotas com uma nova compilação e remover frotas antigas conforme necessário. Para obter mais informações, consulte [Gerenciando o posicionamento da sessão de jogo com Amazon GameLift Servers filas](#).
- Use aliases a fim de transferir jogadores para uma nova compilação de jogo. Ao enviar solicitações de sessão de jogo para Amazon GameLift Servers, especifique um alias de frota em vez de um ID de frota. Para obter mais informações, consulte [Crie um Amazon GameLift Servers alias](#).

- Configure para desenvolvimento iterativo. Durante o desenvolvimento do jogo, explore as opções para configurar um ambiente de teste hospedado que ofereça suporte ao rápido desenvolvimento iterativo. Consulte [Configure para desenvolvimento iterativo com Amazon GameLift Servers Anywhere](#).

Experimente esses recursos do [Amazon GameLift Servers Kit de ferramentas](#) no Github:

Ferramenta de atualização rápida de compilação (somente para desenvolvimento)

Essa ferramenta ajuda você a modificar compilações de servidores de jogos que já estão implantadas em computadores em uma EC2 frota gerenciada, economizando tempo durante a rápida iteração de desenvolvimento. A ferramenta oferece várias opções: você pode substituir a compilação inteira do jogo ou alterar arquivos específicos, além de gerenciar como reiniciar os processos do servidor de jogos após as atualizações. Também é possível usá-la para atualizar todas os recursos de computação em uma frota ou direcionar computações individuais.

Visite o Amazon GameLift Servers Repositório do kit de ferramentas no Github para obter a [ferramenta de atualização rápida de compilação](#) no Github e aprender mais sobre como usá-la.

Exemplo de script de implantação de produção

Este script ilustra como você pode automatizar o processo de atualização de compilações de servidores de jogos que são implantadas em frotas EC2 gerenciadas em produção. Para usar esse script, seu Amazon GameLift Servers a solução de hospedagem deve usar aliases para abstrair a frota IDs. O exemplo de script lida com as seguintes etapas básicas: fazer upload de uma compilação atualizada, criar uma compilação e implantá-la em uma nova frota, redirecionar o tráfego de jogadores de uma frota existente para a nova frota e excluir a frota antiga. Personalize o exemplo de script para atender aos seus requisitos específicos de implantação.

Visite o Amazon GameLift Servers Repositório do kit de ferramentas no Github para obter o [exemplo de script de implantação de produção](#) no Github e aprender mais sobre como usá-lo.

Crie uma imagem de contêiner para Amazon GameLift Servers

Este tópico descreve como criar uma imagem de contêiner com seu software de servidor de jogos para usar com Amazon GameLift Servers. Uma imagem de contêiner de servidor de jogos inclui o executável do servidor de jogos e todas as dependências necessárias para execução. As imagens de contêiner do servidor de jogos são usadas com uma solução de hospedagem de contêineres

Amazon GameLift Servers gerenciados. Para obter detalhes sobre como criar a solução completa, consulte:

- [Roteiro de desenvolvimento para hospedagem com contêineres Amazon GameLift Servers gerenciados](#)
- [Como os contêineres funcionam em Amazon GameLift Servers](#)

Conclua as tarefas a seguir para preparar a imagem do contêiner do servidor de jogos para implantação em uma frota de Amazon GameLift Servers contêineres. Antes de iniciar essas tarefas, conclua a integração do código do servidor do jogo com o SDK do Amazon GameLift Servers servidor.

Tópicos

- [Crie uma imagem de contêiner de servidor de jogos](#)
- [Envie uma imagem de contêiner para o Amazon ECR](#)

Crie uma imagem de contêiner de servidor de jogos

Siga estas instruções ao trabalhar em uma plataforma baseada em Linux ou ao usar o Windows Subsystem for Linux (WSL) com o Docker instalado.

Para criar uma imagem de contêiner de servidor de jogos

1. Prepare um diretório de trabalho com seu software de servidor de jogos. Em uma máquina local, crie um diretório de trabalho para organizar os arquivos do contêiner do servidor de jogos. Sua imagem de contêiner usa essa estrutura de arquivos ao implantar o contêiner em Amazon GameLift Servers recursos para hospedagem. Por exemplo:

```
[~/]$ mkdir -p work/glc/gamebuild && cd work && find .  
.  
./glc  
./glc/gamebuild
```

Note

Se você está testando esse recurso e ainda não tem um servidor de jogo em funcionamento, experimente nosso servidor de jogos de amostra [SimpleServer](#), que está disponível em GitHub.

2. Crie um novo Dockerfile usando o modelo fornecido.
3. Siga as instruções no modelo do Dockerfile para atualizá-lo para seu próprio uso.
 - Atualize a imagem base conforme necessário.
 - Defina variáveis de ambiente para a construção do seu servidor de jogos.
4. Crie a imagem do contêiner. Execute `docker build`, especificando o nome do seu próprio repositório. Por exemplo:

```
[~/work/glc]$ docker build -t <local repository name>:<optional tag> .
```

Você pode visualizar seus repositórios e imagens IDs usando o `docker images` comando, conforme ilustrado neste exemplo:

Modelo de Dockerfile para uma imagem de contêiner de servidor de jogos

Este modelo contém as instruções mínimas que um contêiner de servidor de jogo precisa para ser usado em uma Amazon GameLift Servers frota. Modifique o conteúdo conforme necessário para seu servidor de jogos.

```
# 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"]
```

Envie uma imagem de contêiner para o Amazon ECR

Depois de criar uma imagem de contêiner para implantação Amazon GameLift Servers, armazene a imagem em um repositório público ou privado no Amazon ECR. Esse repositório recebe um valor de URI, que é Amazon GameLift Servers usado para tirar um instantâneo da imagem para implantação em uma frota de contêineres.

Note

Se você ainda não tem um repositório privado do Amazon ECR, [crie um](#).

Para enviar sua imagem de contêiner para o Amazon ECR

1. Obtenha suas credenciais do Amazon ECR. Antes de enviar uma imagem de contêiner para o Amazon ECR, primeiro adquira suas AWS credenciais de forma temporária e forneça-as ao Docker. O Docker precisa dessas credenciais para fazer login.

```
[~/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. Copie o URI do [repositório privado do Amazon ECR](#) que você deseja usar.
3. Aplique uma tag Amazon ECR à imagem do seu contêiner.

Example

```
[~/work/glc]$ docker tag <IMAGE ID from above> <Amazon ECR private repository  
URI>:<optional tag>
```

4. Envie sua imagem de contêiner para o Amazon ECR

Example

```
[~/work/glc]$ docker image push <Amazon ECR private repository URI>
```

Implante um script para Amazon GameLift ServersRealtime

Quando você estiver pronto Amazon GameLift Servers Realtime para implantar seu jogo, faça upload dos arquivos de script de Realtime servidor completos para Amazon GameLift Servers. Faça isso criando um recurso de Amazon GameLift Servers script e especificando a localização dos seus arquivos de script. Também é possível atualizar os arquivos de script de servidores que já estiverem implantados carregando novos arquivos para um recurso de script existente.

Ao criar um novo recurso de script, Amazon GameLift Servers atribui a ele uma ID de script exclusiva (por exemplo, `script-1111aaaa-22bb-33cc-44dd-5555eeee66ff`) e carrega uma cópia dos arquivos de script. O tempo de upload depende do tamanho de seus arquivos de script e da velocidade da sua conexão.

Depois de criar o recurso de script, Amazon GameLift Servers implanta o script com uma nova Amazon GameLift Servers Realtime frota. Amazon GameLift Servers instala o script do servidor em cada instância da frota, inserindo os arquivos de script em `/local/game`.

Para solucionar problemas de ativação da frota relacionados ao script de servidores, consulte [Depurar problemas de frota do Amazon GameLift Servers](#).

Empacotar os arquivos de script

Seu script de servidores pode incluir um ou mais arquivos combinados em um único arquivo .zip para upload. O arquivo .zip deve conter todos os arquivos que o script precisa para ser executado.

É possível armazenar os arquivos de script compactados em um diretório de arquivos local ou em um bucket do Amazon Simple Storage Service (Amazon S3).

Carregar arquivos de script em um diretório local

Se você tiver seus arquivos de script armazenados localmente, poderá enviá-los Amazon GameLift Servers de lá. Para criar o recurso de script, use o Amazon GameLift Servers console ou o [AWS Command Line Interface \(AWS CLI\)](#).

GameLift Servidores Amazon console

Para criar um recurso de script

1. Abra o [console de Amazon GameLift Servers](#).
2. No painel de navegação, escolha Hospedagem, Scripts.
3. Na página Scripts, escolha Criar script.
4. Na página Criar script, em Configurações do script, faça o seguinte:
 - a. Para Nome, insira um nome de script.
 - b. (Opcional) Para Versão, insira informações da versão. Como o conteúdo de um script pode ser atualizado, os dados da versão podem ser úteis para rastrear atualizações.
 - c. Para Origem do script, escolha Carregar um arquivo .zip.
 - d. Para Arquivos de script, escolha Escolher arquivo, procure o arquivo .zip que contém o script e, em seguida, escolha esse arquivo.
5. (Opcional) Em Tags, adicione tags ao script inserindo pares de Chave e Valor.
6. Escolha Criar.

Amazon GameLift Servers atribui uma ID ao novo script e carrega o arquivo.zip designado. É possível visualizar o novo script, incluindo seu status, na página Scripts.

AWS CLI

Use o comando [create-script](#) AWS CLI para definir o novo script e fazer upload dos arquivos de script de servidor.

Para criar um recurso de script

1. Coloque o arquivo .zip em um diretório onde você possa usar a AWS CLI.
2. Abra uma janela de linha de comando e alterne para o diretório onde você colocou o arquivo .zip.
3. Insira os seguintes parâmetros e o comando create-script. Para o parâmetro --zip-file, certifique-se de adicionar a string fileb:// ao nome do arquivo .zip. Ele identifica o arquivo como binário para Amazon GameLift Servers processar o conteúdo compactado.

```
aws gamelift create-script \  
  --name user-defined name of script \  
  --zip-file fileb://
```

```
--script-version user-defined version info \  
--zip-file fileb://name of zip file \  
--region region name
```

Exemplo

```
aws gamelift create-script \  
  --name "My_Realtime_Server_Script_1" \  
  --script-version "1.0.0" \  
  --zip-file fileb://myrealtime_script_1.0.0.zip \  
  --region us-west-2
```

Em resposta à sua solicitação, Amazon GameLift Servers retorna o novo objeto de script.

4. Para ver o novo script, chame [describe-script](#).

Carregue arquivos de script no Amazon S3

Você pode armazenar seus arquivos de script em um bucket do Amazon S3 e carregá-los Amazon GameLift Servers de lá. Ao criar seu script, você especifica a localização do bucket do S3 e Amazon GameLift Servers recupera seus arquivos de script do Amazon S3.

Para criar um recurso de script

1. Armazene os arquivos de script em um bucket do S3. Crie um arquivo.zip contendo os arquivos de script do servidor e carregue-o em um bucket do S3 em uma Conta da AWS que você controla. Anote o URI do objeto — você precisa disso ao criar um Amazon GameLift Servers script.

Note

Amazon GameLift Servers não oferece suporte ao upload de buckets do S3 com nomes que contenham um ponto (.).

2. Conceda ao Amazon GameLift Servers acesso aos arquivos de script. Para criar uma função AWS Identity and Access Management (IAM) que permita Amazon GameLift Servers acessar o bucket do S3 contendo seu script de servidor, siga as instruções em [Configure uma função de serviço do IAM para Amazon GameLift Servers](#). Depois de criar a nova função, anote seu nome, que é necessário ao criar um script.

3. Crie um script. Use o Amazon GameLift Servers console ou o AWS CLI para criar um novo registro de script. Para fazer essa solicitação, você precisa ter a permissão `PassRole` do IAM, conforme descrito em [Exemplos de permissão do IAM para Amazon GameLift Servers](#).

GameLift Servidores Amazon console

1. No [Amazon GameLift Servers console](#), no painel de navegação, escolha Hospedagem, Scripts.
2. Na página Scripts, escolha Criar script.
3. Na página Criar script, em Configurações do script, faça o seguinte:
 - a. Para Nome, insira um nome de script.
 - b. (Opcional) Para Versão, insira informações da versão. Como o conteúdo de um script pode ser atualizado, os dados da versão podem ser úteis para rastrear atualizações.
 - c. Para Origem do script, escolha URI do Amazon S3.
 - d. Insira o URI do S3 do objeto de script que carregado no Amazon S3 e, em seguida, escolha a versão do objeto. Se você não se lembra do URI e da versão do objeto do Amazon S3, escolha Procurar S3 e, em seguida, pesquise o objeto de script.
4. (Opcional) Em Tags, adicione tags ao script inserindo pares de Chave e Valor.
5. Escolha Criar.

Amazon GameLift Servers atribui uma ID ao novo script e carrega o arquivo.zip designado. É possível visualizar o novo script, incluindo seu status, na página Scripts.

AWS CLI

Use o comando [create-script](#) AWS CLI para definir o novo script e fazer upload dos arquivos de script de servidor.

1. Abra uma janela de linha de comando e alterne para um diretório em que você possa usar a AWS CLI.
2. Insira os seguintes parâmetros e o comando `create-script`. O parâmetro de `--storage-location` especifica o local do bucket do Amazon S3 dos arquivos de script.

```
aws gamelift create-script \  
  --name [user-defined name of script] \  
  --storage-location [storage location]
```

```
--script-version [user-defined version info] \  
--storage-location "Bucket"=S3 bucket name,"Key"=name of zip file in S3 bucket,"RoleArn"=Access role ARN \  
--region region name
```

Exemplo

```
aws gamelift create-script \  
  --name "My_Realtime_Server_Script_1" \  
  --script-version "1.0.0" \  
  --storage-location "Bucket"="gamelift-script","Key"="myrealtime_script_1.0.0.zip","RoleArn"="arn:aws:iam::123456789012:role/S3Access" \  
  --region us-west-2
```

Em resposta à sua solicitação, Amazon GameLift Servers retorna o novo objeto de script.

3. Para visualizar o novo script, chame [describe-script](#).

Atualizar arquivos de Amazon GameLift ServersRealtime script

Você pode atualizar os metadados de um recurso de script usando o Amazon GameLift Servers console ou o [update-script](#) AWS CLI comando.

Você também pode atualizar o conteúdo do script para um recurso de script. Amazon GameLift Servers implanta conteúdo de script em todas as instâncias da frota que usam o recurso de script atualizado. Quando o script atualizado é implantado, as instâncias o usam ao iniciar novas sessões de jogo. As sessões de jogo que já estão em execução no momento da atualização não usam o script atualizado.

Para atualizar arquivos de script

- Para arquivos de script armazenados localmente, para carregar o arquivo.zip de script atualizado, use o Amazon GameLift Servers console ou o `update-script` comando.
- Para arquivos de script armazenados em um bucket do Amazon S3, faça o upload dos arquivos de script atualizados para o bucket do S3. Amazon GameLift Servers verifica periodicamente os arquivos de script atualizados e os recupera diretamente do bucket do S3.

Configurando uma frota de hospedagem com Amazon GameLift Servers

Nesta seção, você encontrará informações sobre como projetar, construir e manter Amazon GameLift Servers frotas para hospedar seus servidores de jogos. Consulte [Amazon GameLift Servers opções de hospedagem](#) para saber mais sobre as soluções de hospedagem Amazon GameLift Servers oferecidas, incluindo aquelas que usam EC2 frotas gerenciadas, frotas autogerenciadas do Anywhere para seu hardware local e uma solução híbrida que usa ambas.

Tópicos

- [Características da frota](#)
- [Como funciona a criação de Amazon GameLift Servers frotas](#)
- [Amazon GameLift Servers EC2 frotas gerenciadas](#)
- [Amazon GameLift Servers frotas de contêineres gerenciadas](#)
- [Amazon GameLift Servers Frotas Anywhere](#)
- [Abstrato e Amazon GameLift Servers designação de frota com um alias](#)

Características da frota

Uma Amazon GameLift Servers frota é uma coleção de recursos de computação que administram seus servidores de jogos e hospedam sessões de jogo para jogadores. As frotas podem variar de acordo com o tipo de recursos computacionais que você usa e como a frota é gerenciada. O tamanho de uma frota (o número de sessões de jogo e jogadores que ela pode suportar) depende da quantidade de recursos computacionais que você fornece. Todas as Amazon GameLift Servers frotas têm as seguintes características:

- Os processos do servidor de jogos que são executados em todas as frotas são integrados ao SDK do servidor Amazon GameLift Servers e se comunicam com o Amazon GameLift Servers serviço da mesma maneira. Os servidores de jogos relatam sua disponibilidade para hospedar sessões de jogo e jogadores, responder às solicitações para iniciar ou interromper as sessões de jogo e outras interações.
- Amazon GameLift Servers lida com o posicionamento das sessões de jogo para todas as frotas da mesma forma. Amazon GameLift Servers acompanha o status do servidor de jogo de uma frota e escolhe entre os servidores de jogos disponíveis para hospedar uma nova sessão de jogo. Esse

processo é usado, quer seu jogo coloque sessões de jogo em uma única frota ou use uma [fila de sessões de jogo](#) para equilibrar a hospedagem em várias frotas. Com uma fila, você também pode personalizar as decisões de posicionamento para considerar fatores como custo de recursos e latência.

- Todas as frotas suportam o uso de um FlexMatch matchmaker em colaboração com uma fila de colocação de sessões de jogo. O Amazon GameLift Servers serviço recebe solicitações de partidas de jogadores, forma as partidas e as passa para a fila da sessão de jogo para encontrar os servidores de jogos disponíveis.
- Amazon GameLift Servers coleta uma ampla variedade de métricas de frota. Isso inclui métricas de status para recursos de computação e processos de servidor, bem como métricas de uso de sessões de jogos e atividades de jogadores. Consulte a lista completa de métricas disponíveis em [Monitore Amazon GameLift Servers com a Amazon CloudWatch](#).

Como funciona a criação de Amazon GameLift Servers frotas

Quando você solicita uma nova frota, Amazon GameLift Servers inicia um fluxo de trabalho para criar o recurso de frota. Ao concluir cada etapa do fluxo de trabalho, Amazon GameLift Servers atualiza o status da frota e emite uma série de eventos para comunicar o progresso na criação da frota.

Amazon GameLift Servers usa dois tipos de eventos. Os eventos de transição do estado da frota marcam quando o status da frota muda. Os eventos de criação de frotas fornecem marcadores adicionais para ajudar na depuração de problemas de criação. Você pode acompanhar todos os eventos usando o Amazon GameLift Servers console ou chamando a operação Amazon GameLift Servers da API [DescribeFleetEvents](#). Você também pode rastrear o status da frota e da localização usando [DescribeFleetAttributes](#) ou [DescribeFleetLocationAttributes](#).

Amazon GameLift Servers EC2 frotas gerenciadas

Amazon GameLift Servers EC2 frotas gerenciadas oferecem recursos baseados em nuvem para hospedagem de produção com Amazon GameLift Servers. Com uma frota gerenciada, você obtém a flexibilidade, a segurança e a confiabilidade de Nuvem AWS recursos que são ainda mais otimizados para hospedagem de jogos multijogador. A ferramenta Amazon GameLift Servers o serviço fornece ferramentas robustas de gerenciamento de host.

Uma EC2 frota gerenciada é um conjunto de computadores virtuais que Amazon GameLift Servers possui e opera em seu nome e com base em suas opções de configuração. Computações são instâncias do Amazon Elastic Compute Cloud EC2 (Amazon) que estão fisicamente localizadas em

Regiões da AWS ou em Zonas Locais. Ao criar uma frota, você escolhe um tipo de EC2 instância para seus computadores com base na capacidade de computação, memória, armazenamento, recursos de rede e outros fatores.

Com uma EC2 frota gerenciada, você carrega sua compilação de servidor de jogo para Amazon GameLift Servers. Quando você cria uma frota, o serviço implanta sua compilação nos computadores da frota e inicia os processos do servidor de jogos. Cada processo de servidor de jogo lançado estabelece uma conexão com o Amazon GameLift Servers serviço e relata a prontidão para sediar uma sessão de jogo.

Além da implantação da frota, Amazon GameLift Servers processa as seguintes tarefas de gerenciamento do host para que você não precise:

- Monitora o status de todas os recursos de computação da frota e substitui as computações obsoletas ou não íntegras.
- Lida com a autenticação para comunicação entre os processos do servidor e o Amazon GameLift Servers serviço.
- Inicia e interrompe automaticamente os processos do servidor de jogos em cada computação, com base na sua configuração de tempo de execução.
- Oferece ferramentas de auto-scaling para ajustar dinamicamente a capacidade da frota para atender à demanda dos jogadores.
- Relata métricas de desempenho para as EC2 instâncias da frota.

Veja estes tópicos sobre como configurar e manter EC2 frotas gerenciadas:

- [Roteiro de desenvolvimento para hospedagem com gerenciamento Amazon GameLift Servers EC2](#)
- [Crie uma EC2 frota Amazon GameLift Servers gerenciada](#)
- [Escalando a capacidade de hospedagem de jogos com Amazon GameLift Servers](#)
- [Personalize seu Amazon GameLift Servers EC2 frotas gerenciadas](#)
- [Atualizar uma configuração de Amazon GameLift Servers frota](#)

Fluxo de trabalho de criação de EC2 frotas

Para frotas gerenciadas, Amazon GameLift Servers configura o recurso da frota e também implanta um conjunto de recursos computacionais com o software do servidor de jogos instalado e em execução. Quando o fluxo de trabalho de criação é concluído e bem-sucedido, a frota tem uma

EC2 instância ativa na região de origem da frota e uma em cada local remoto da frota. Todas as instâncias têm jogos prontos para hospedar sessões de jogo.

1. Amazon GameLift Servers cria o recurso da frota na região de origem da frota e define a capacidade desejada em cada local em uma (1) instância. O status da frota e da localização está definido como Novo.
2. Amazon GameLift Servers começa a gravar eventos no registro de eventos da frota.
3. Amazon GameLift Servers define o status da frota como Baixando e começa a preparar o software do servidor de jogos para implantação.
 - a. Obtém a compilação do servidor de jogo carregada e extrai os arquivos compactados.
 - b. Executa scripts de instalação, se fornecidos.
 - c. Define o status da frota como Validando e começa a verificar se nenhum erro ocorreu ao baixar e instalar os arquivos de compilação.
4. Amazon GameLift Servers define o status da frota como Building, configura o hardware da frota e aloca uma EC2 instância para cada instância da frota.
5. Amazon GameLift Servers define o status da frota como Ativando. Inicia um processo de servidor de jogo em cada instância (com base nas instruções de tempo de execução da frota) e testa a conectividade entre a compilação e o Amazon GameLift Servers serviço.
6. Quando os processos do servidor de jogos em cada instância estabelecem uma conexão e relatam a prontidão para hospedar sessões de jogo, Amazon GameLift Servers define os status da frota e da localização como Ativo. Nesse ponto, a frota é considerada pronta para hospedar sessões de jogo.

Crie uma EC2 frota Amazon GameLift Servers gerenciada

Este tópico descreve como criar uma EC2 frota Amazon GameLift Servers gerenciada. As frotas gerenciadas usam instâncias de computação do Amazon Elastic Compute Cloud EC2 (Amazon) que são otimizadas para hospedagem de jogos multijogador. Você pode criar frotas gerenciadas que implantam computações em Zonas Locais Regiões da AWS e Globalmente suportadas por. Amazon GameLift Servers

Quando você cria uma nova EC2 frota gerenciada, o processo de criação da frota começa imediatamente. Uma frota gerenciada passa por várias fases enquanto Amazon GameLift Servers prepara a criação do servidor de jogos, implanta EC2 instâncias com a compilação instalada e inicia servidores de jogos em cada instância. Você pode monitorar o status de uma frota no console ou

usando o `uring` AWS Command Line Interface (AWS CLI). Uma frota está pronta para hospedar sessões de jogo quando seu status atinge `ACTIVE`. Para obter mais informações sobre a criação de frotas gerenciadas, consulte os seguintes tópicos:

- [Como funciona a criação de Amazon GameLift Servers frotas](#)
- [Depurar problemas de frota do Amazon GameLift Servers](#)

Para criar uma EC2 frota gerenciada

Use o Amazon GameLift Servers console ou o AWS Command Line Interface (AWS CLI) para criar uma EC2 frota gerenciada.

Console

No [Amazon GameLift Servers console](#), use o painel de navegação para abrir a página `Fleets`. Escolha `Criar frota` para iniciar o fluxo de trabalho de criação da frota.

Etapa 1 Escolha o tipo de computação

Selecione a EC2 opção `Gerenciado` e escolha `Avançar`.

Etapa 2 Defina os detalhes da frota

Nesta etapa, especifique algumas configurações para toda a frota.

 Para uma configuração mínima da frota:

- Forneça o nome da frota.
- Escolha um tipo binário e especifique uma compilação ou script carregado.
- Ignore as seções sobre detalhes e tags adicionais.

1. Preencha a seção Detalhes da frota:

- a. Insira o Nome de uma frota. Recomendamos usar um padrão de nomenclatura de frota que facilite a identificação dos tipos de frota ao visualizar listas de frotas.
- b. Forneça uma breve Descrição da frota.
- c. Em Tipo binário, selecione `Compilar` para indicar que você está implantando uma versão de servidor de jogo personalizada ou selecione `Script` se estiver implantando

nessa Amazon GameLift Servers Realtime frota. Selecione uma compilação ou um script carregado na lista suspensa.

2. (Opcional) Defina Detalhes adicionais conforme necessário.

- a. Se o executável do seu servidor de jogos precisar acessar outros AWS recursos em sua conta, especifique uma função de instância do IAM com as permissões necessárias. Para obter mais informações, incluindo como autorizar outros aplicativos do lado do servidor (como CloudWatch agente), consulte [Comunique-se com outros AWS recursos de suas frotas](#). Essa configuração não pode ser alterada após a criação da frota.

É necessário criar o perfil antes de criar uma frota que o utilize. Além disso, para criar uma frota com um perfil de instância, o usuário da AWS precisa ter permissão `PassRole` do IAM (consulte [Exemplos de permissão do IAM para Amazon GameLift Servers](#)).

- b. Ative a opção Gerar um certificado TLS para configurar a autenticação e a criptografia do jogo. Os clientes do jogo usam esse certificado para autenticar um servidor de jogos ao se conectar e criptografar toda a comunicação entre cliente e servidor. Para cada instância em uma frota habilitada para TLS, o Amazon GameLift Servers também cria uma nova entrada DNS com o certificado. Essa configuração não pode ser alterada após a criação da frota.
 - c. Se você quiser combinar dados métricos dessa frota e de outras, especifique um nome de Grupo de métricas. Use o mesmo nome de grupo de métricas para todas as frotas que deseja combinar. Visualize as métricas do grupo de métricas para ver os dados agregados.
3. (Opcional) Adicione tags ao recurso da frota. Cada tag consiste de uma chave e um valor opcional, que podem ser definidos. Atribua tags aos AWS recursos que você deseja categorizar de maneiras úteis, como por finalidade, proprietário ou ambiente. Escolha Adicionar nova tag para cada tag a ser adicionada.
4. Escolha Avançar para continuar o fluxo de trabalho.

Etapa 3 Defina detalhes da instância

Nessa etapa, especifique o tipo de recursos de hospedagem a serem usados e onde você deseja implantá-los. Ao escolher vários locais, é possível implantar o servidor de jogos em

uma localização geográfica mais ampla, o que o coloca mais perto dos jogadores e minimiza a latência. Nem todos os tipos de EC2 instância estão disponíveis em todos os locais.

 Para uma configuração mínima da frota:

- Não adicione locais remotos.
- Defina o tipo de frota como “Sob demanda”. As frotas spot exigem trabalho adicional de configuração.
- Defina o tipo de instância como “c5.large”. Esse tipo de instância comumente usado está disponível em todos Regiões da AWS.

1. Em Implantação de instâncias, especifique os locais e o tipo da frota.
 - a. Selecione um ou mais Locais adicionais nos quais você deseja implantar instâncias de frota. Esses locais remotos são adicionados ao local de origem da frota (que é pré-selecionado), que é Região da AWS onde você está criando essa frota. Você pode selecionar locais remotos de todas Regiões da AWS as Zonas Locais Amazon GameLift Servers compatíveis.

Para saber mais sobre os locais compatíveis, incluindo como usar um Região da AWS que não está habilitado por padrão, consulte [Amazon GameLift Servers locais de serviço](#) Hospedagem gerenciada. Analise também Amazon GameLift Servers [as cotas](#) de locais por frota.

- b. Escolha usar instâncias Sob demanda ou Spot para essa frota. Para obter informações sobre tipos de frotas, consulte [Instâncias sob demanda versus instâncias spot](#).
2. Escolha uma configuração de EC2 instância da Amazon que atenda às suas necessidades e esteja disponível em todos os locais selecionados. Essa lista é filtrada com base no local atual e nas seleções do tipo de frota. É possível filtrá-la ainda mais por outros fatores, como a família e a arquitetura do tipo de instância. Após criar a frota, não será possível alterar o tipo de instância.

Alguns locais têm opções limitadas de tipo de instância. Se o tipo de instância de sua preferência não estiver disponível para todos os locais, escolha o valor da disponibilidade do local para ver os detalhes completos. Para acomodar todos os locais, talvez seja necessário criar frotas separadas com diferentes tipos de instância.

Para obter mais informações sobre a escolha de um tipo de instância, consulte [Tipos de instância](#). Para saber mais sobre as arquiteturas Amazon EC2 Arm, consulte [Processador AWS Graviton](#) e tipos de [EC2 instância da Amazon](#). Para ver uma lista completa dos tipos de instância compatíveis com Amazon GameLift Servers, consulte a referência da API para [EC2InstanceType](#)(CreateFleet()).

 Note

As instâncias do Graviton Arm exigem um Amazon GameLift Servers servidor baseado no sistema operacional Linux. O Server SDK 5.1.1 ou mais recente é necessário para C++ e C#. O Server SDK 5.0 ou mais recente é necessário para o Go. Essas instâncias não oferecem out-of-the-box suporte para a instalação do Mono no Amazon Linux 2023 (AL2023) ou no Amazon Linux (AL22).

3. Escolha Avançar para continuar o fluxo de trabalho.

Etapa 4 Configure o tempo de execução

Nesta etapa, descreva como você deseja que cada instância da frota execute o software do servidor de jogos. Defina um item de linha de processo do servidor separado para cada executável a ser executado em uma instância e decida quantos processos de cada servidor serão executados simultaneamente. Abra portas em cada instância para permitir que os jogadores se conectem diretamente aos servidores de jogos. Você pode atualizar essas configurações da frota a qualquer momento.

 Para uma configuração mínima da frota:

- Defina um único item de linha de processo do servidor para o executável do servidor de jogos. Se o servidor de jogos exigir que outros processos estejam em execução, crie também uma definição para cada um deles.
- Use o número padrão de processos simultâneos (1) para cada item de linha.
- Ignore as configurações de ativação da sessão de jogo.
- Especifique um número de porta única.
- Ignore as configurações de recursos da sessão de jogo.

1. Crie uma configuração de tempo de execução para instruir Amazon GameLift Servers sobre como executar os processos do servidor em cada instância da frota. Você pode alterar a configuração de tempo de execução de uma frota a qualquer momento após a implantação.
 - a. Insira o Caminho de inicialização para um arquivo executável na compilação. Em instâncias do Windows, os executáveis do servidor de jogo são criados no caminho `C:\game`. Em instâncias do Linux, os servidores de jogos são criados para `/local/game`. Exemplos: **`C:\game\MyGame\server.exe`**, **`/local/game/MyGame/server.exe`** ou **`MyRealtimeLaunchScript.js`**.
 - b. Insira os Parâmetros de inicialização opcionais para passar ao executável do jogo. Exemplo: **`+sv_port 33435 +start_lobby`**.
 - c. Especifique o número de Processos simultâneos a serem executados em cada instância. Para um executável de servidor de jogos, cada processo pode hospedar uma sessão de jogo, então os processos simultâneos determinam o número de sessões de jogo que a instância pode hospedar simultaneamente.

Analise as Amazon GameLift Servers [cotas](#) nos processos do servidor por instância. Essas cotas se aplicam ao total de processos simultâneos para todas as configurações. Se você configurar a frota para excedê-las, ela não poderá ser ativada.

2. Use os padrões de Ativação da sessão de jogo ou personalize-os para o jogo. Se a configuração do tempo de execução exigir vários processos simultâneos do servidor de jogos por instância, essas configurações determinarão a rapidez com que novas sessões de jogo podem ser iniciadas.
 - a. Defina a Ativação máxima da sessão de jogos simultâneos para limitar o número de servidores de jogos em uma instância que estão preparando uma nova sessão de jogo. Essa configuração é útil quando a inicialização de várias novas sessões de jogo consome muitos recursos e pode afetar o desempenho de outras sessões de jogo em execução.
 - b. Defina o novo tempo limite de ativação para refletir o tempo máximo que uma nova sessão de jogo deve levar para concluir a ativação e relatar que está pronto para hospedar jogadores. Amazon GameLift Servers encerra a ativação de uma sessão de jogo se ela exceder esse valor.

3. Abra as configurações de EC2 porta para permitir que o tráfego de entrada acesse os processos do servidor na frota. Essas configurações não são necessárias para criar uma frota, mas é necessário defini-las antes que os jogadores possam se conectar às sessões de jogo na frota.

Para cada configuração de porta, escolha o Tipo de protocolo de transferência de dados a ser usado para comunicação entre o cliente do jogo e o servidor do jogo. Forneça um intervalo de portas (no formato nnnnn[-nnnn]) e um intervalo de endereços IP usando a notação CIDR (como 0.0.0.0/0, que permite acesso a qualquer pessoa).

Se você precisar definir vários intervalos não consecutivos, crie várias configurações de porta.

4. Especifique as Configurações de recursos para a sessão de jogo opcionais. Você pode atualizar essas configurações a qualquer momento após a implantação.
 - a. Ative ou desative a política de proteção de escalabilidade do jogo para todas as instâncias da frota. Durante um evento de redução de escala, não Amazon GameLift Servers encerrará instâncias de frota protegidas se elas estiverem hospedando sessões de jogo ativas.
 - b. Defina um Limite de criação de recursos máximo se quiser restringir o número de sessões de jogo que um jogador pode criar durante um período de tempo especificado.
5. Escolha Avançar para continuar o fluxo de trabalho.

Etapa 5 Revise e crie

Revise as configurações antes de criar a frota. Embora algumas configurações possam ser atualizadas posteriormente (consulte [Atualizar uma configuração de Amazon GameLift Servers frota](#)), alterações nas configurações a seguir não são permitidas após a criação da frota:

- Tipo de computação: você não pode converter uma EC2 frota gerenciada em uma frota Anywhere.
- Compilação ou script: para implantar uma atualização na compilação ou no script do servidor de jogos, é necessário criar uma frota.
- Opções adicionais, incluindo perfil de instância e geração de certificado TLS.
- Detalhes da instância, incluindo tipo de frota (spot ou sob demanda) e tipo de EC2 instância.

Quando você estiver pronto para implantar a nova frota, escolha Criar. Amazon GameLift Servers inicia imediatamente o processo de ativação da frota, atribuindo uma ID exclusiva e colocando a frota em NEW status. Acompanhe o progresso da frota na página Frotas. Visualize a página de detalhes da frota e acesse a guia Eventos.

Você pode ajustar a capacidade de hospedagem de uma frota depois que a frota atingir o status ATIVO. Amazon GameLift Servers implanta inicialmente uma frota com uma única instância em cada local da frota, e você ajusta a capacidade adicionando instâncias a cada local. Para obter mais informações, consulte [Escalação a capacidade de hospedagem de jogos com Amazon GameLift Servers](#).

AWS CLI

Use o [create-fleet](#) comando para criar uma frota do tipo EC2 computacional. Amazon GameLift Servers cria o recurso de frota em seu padrão atual Região da AWS (ou você pode adicionar uma tag `--region` para especificar uma diferente Região da AWS).

Criar uma frota gerenciada mínima

O exemplo de solicitação a seguir cria uma frota com as configurações mínimas necessárias para implantar uma frota com servidores de jogos em execução aos quais os clientes do jogo podem se conectar. A nova frota tem as seguintes características:

- Ele especifica uma compilação de servidor de jogo, que foi carregada Amazon GameLift Servers e está em READY status.
- Utiliza instâncias sob demanda `c5.large` com um sistema operacional que corresponde à compilação de jogo selecionada.
- Ele define a sede Região da AWS da frota `us-west-2` e implanta instâncias somente nessa região.
- Com base na configuração do tempo de execução, cada computação da frota executa um processo de servidor de jogos, o que significa que cada computação pode hospedar somente uma sessão de jogo por vez. O tempo limite de ativação da sessão de jogo é definido como o valor padrão de 300 segundos, e não há limite para o número de ativações simultâneas.
- Os jogadores podem se conectar aos servidores de jogos usando uma configuração de porta única de 33435.
- Todos os outros recursos estão desativados ou usam as configurações padrão.

```
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"
```

Criar uma frota gerenciada totalmente configurada

O exemplo de solicitação a seguir cria uma frota de produção com configurações para todos os recursos opcionais. A nova frota tem as seguintes características:

- Ele especifica uma compilação de servidor de jogo, que foi carregada Amazon GameLift Servers e está em READY status.
- Utiliza instâncias sob demanda c5.large com o sistema operacional que corresponde à compilação de jogo selecionada.
- Ele define a sede Região da AWS da frota us-west-2 e implanta instâncias na região de origem e em um local sa-east-1 remoto.
- Com base na configuração do tempo de execução:
 - Cada computador da frota executa 10 processos de servidor de jogos com os mesmos parâmetros de inicialização, o que significa que cada computação pode hospedar até 10 sessões de jogo simultaneamente.
 - Em cada computação, somente duas sessões de jogo podem ser ativadas ao mesmo tempo. A ativação das sessões de jogo deve estar pronta para hospedar jogadores em um prazo de 300 segundos (5 minutos) ou ser encerrada.
- Os jogadores podem se conectar aos servidores de jogos usando uma porta no seguinte intervalo: 33435 to 33535.
- Gera um certificado TLS para comunicação criptografada entre o cliente e o servidor do jogo.
- Todas as sessões de jogo na frota têm a proteção de sessão de jogo ativada.
- Jogadores individuais são limitados a criar três sessões de jogo dentro de um período de 15 minutos.

- As métricas dessa frota estão incluídas no grupo de métricas AMERfleets, que (neste exemplo) agrega métricas para um grupo de frotas nas Américas do Norte, Central e do Sul.

```
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 a solicitação de criação de frota for bem-sucedida, Amazon GameLift Servers retornará um conjunto de atributos da frota que inclui as configurações solicitadas e uma nova ID da frota. Amazon GameLift Serversem seguida, inicia o processo de ativação da frota e define o status da frota e o status da localização como Novo. Você pode acompanhar o status da frota e visualizar outras informações da frota usando estes comandos da 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](#)

É possível alterar a capacidade da frota e outras definições de configuração, conforme necessário, usando estes comandos:

- [update-fleet-attributes](#)
- [update-fleet-capacity](#)
- [update-fleet-port-settings](#)
- [update-runtime-configuration](#)
- [create-fleet-locations](#)
- [delete-fleet-locations](#)

Personalize seu Amazon GameLift Servers EC2 frotas gerenciadas

Os tópicos desta seção descrevem algumas das personalizações que você pode fazer ao criar uma solução de hospedagem com Amazon GameLift Servers EC2 frotas gerenciadas. Eles fornecem orientação e práticas recomendadas para criar e configurar uma frota para hospedar o jogo.

As decisões necessárias incluem:

- Onde você deve implantar recursos de hospedagem? A latência é um fator importante na seleção dos locais da frota, mas o custo também varia de acordo com o local.
- Qual tipo de EC2 instância oferecerá melhor suporte ao seu jogo? Escolha entre os tipos de instância que estão disponíveis em todos os locais da frota para usar a melhor combinação de arquitetura de computação, memória, armazenamento e capacidade de rede.
- Qual o tamanho do tipo de instância que você precisa? Escolha o tamanho do tipo de instância com base nos requisitos de recursos (memória e CPU) do software do servidor de jogos e em outros fatores.
- A frota deve usar instâncias spot ou sob demanda? Considere se você pode aproveitar os preços spot mais baixos, considerando como Amazon GameLift Servers protege contra a chance de interrupções nas sessões de jogo.
- Como você quer que o software de servidor de jogos seja executado em cada instância da frota? A configuração do tempo de execução diz Amazon GameLift Servers qual software de servidor executar e como.

Tópicos

- [Escolher recursos de computação para uma frota gerenciada](#)

- [Gerencie como Amazon GameLift Servers lança servidores de jogos](#)

Escolher recursos de computação para uma frota gerenciada

Para implantar seus servidores de jogos e hospedar sessões de jogos na nuvem, Amazon GameLift Servers fornece frotas gerenciadas que usam [recursos do Amazon Elastic Compute Cloud EC2 \(Amazon\)](#) chamados de instâncias. Use os tópicos a seguir para ajudar a decidir que tipo de EC2 instâncias você deseja usar para sua solução de hospedagem gerenciada e como configurá-las para executar seu software de servidor de jogos.

Note

Se você planeja usar recursos de hospedagem de sua propriedade, seja hardware local ou outra hospedagem baseada em nuvem, considere as opções de hospedagem híbrida com Amazon GameLift Servers Em qualquer lugar. Consulte [Configurando uma frota de hospedagem com Amazon GameLift Servers](#).

Tópicos

- [Localização da frota](#)
- [Instâncias sob demanda versus instâncias spot](#)
- [Sistemas operacionais](#)
- [Tipos de instância](#)
- [Cotas de serviço](#)

Localização da frota

Considere as localizações geográficas em que você planeja implantar seus servidores de jogos. A disponibilidade do tipo de instância varia de Região da AWS acordo com a zona local.

Para frotas com vários locais, a disponibilidade e as cotas das instâncias dependem de uma combinação da região de origem da frota e dos locais remotos selecionados. Para obter mais informações sobre localizações de frotas, consulte [Amazon GameLift Servers locais de serviço](#).

Para Amazon GameLift Servers Em qualquer frota, você determina a localização do seu hardware físico. Para obter mais informações sobre localizações personalizadas, consulte [Localizações para Amazon GameLift Servers qualquer lugar](#).

Instâncias sob demanda versus instâncias spot

As instâncias EC2 sob demanda e as instâncias spot da Amazon oferecem o mesmo hardware e desempenho, mas diferem em disponibilidade e custo.

Instâncias sob demanda

Você poderá adquirir uma instância sob demanda quando precisar e mantê-la pelo tempo que desejar. As instâncias sob demanda têm um custo fixo, o que significa que você paga pelo tempo que usá-las. Além disso, não há compromissos de longo prazo.

Instâncias spot

As instâncias spot podem oferecer uma alternativa econômica às instâncias sob demanda, utilizando a capacidade de computação não AWS utilizada. Os preços das Instâncias Spot flutuam com base na oferta e na demanda de cada tipo de instância em cada local. AWS pode interromper as Instâncias Spot sempre que precisar da capacidade de volta. Amazon GameLift Servers usa filas e o FleetIQ algoritmo para determinar AWS se vai interromper uma instância spot, ele coloca a instância em um estado de reciclagem. Então, quando não houver sessões de jogo ativas na instância, Amazon GameLift Servers tenta substituí-lo.

Para obter mais informações sobre como usar as instância spot, consulte [Criar uma fila para instâncias spot](#).

Sistemas operacionais

Amazon GameLift Servers as instâncias oferecem suporte a compilações de servidores de jogos que são executadas no Microsoft Windows ou no Amazon Linux. Quando você carrega uma versão de jogo para Amazon GameLift Servers, especifique o sistema operacional do jogo. Quando você cria uma EC2 frota da Amazon para implantar a versão do jogo, Amazon GameLift Servers configura automaticamente as instâncias com o sistema operacional da compilação. Para obter mais informações sobre os sistemas operacionais de servidores de jogos compatíveis, consulte [Obtenha ferramentas Amazon GameLift Servers de desenvolvimento](#).

Ao usar um Amazon GameLift Servers Em qualquer frota, você pode usar qualquer sistema operacional compatível com seu hardware. Amazon GameLift Servers Em qualquer lugar, as frotas exigem que você implante sua versão de jogo no hardware enquanto usa Amazon GameLift Servers para gerenciar seus recursos em um só lugar.

Tipos de instância

O tipo de instância de uma EC2 frota da Amazon determina o tipo de hardware que as instâncias usam. Diferentes tipos de instância oferecem combinações diferentes de capacidade de computação, memória, armazenamento e recursos de rede.

Ao escolher entre os tipos de instância disponíveis para seu jogo, considere:

- A arquitetura computacional do seu servidor de jogos: x64 ou Arm (AWS Graviton).

Note

As instâncias do Graviton Arm exigem um Amazon GameLift Servers servidor construído no sistema operacional Linux. O Server SDK 5.1.1 ou mais recente é necessário para C++ e C#. O Server SDK 5.0 ou mais recente é necessário para o Go. Essas instâncias não oferecem out-of-the-box suporte para a instalação do Mono no Amazon Linux 2023 (AL2023) ou no Amazon Linux (AL22).

- Os requisitos de computação, memória e armazenamento da versão do seu servidor de jogos.
- O número de processos do servidor que você planeja executar por instância.

Ao usar um tipo de instância maior, você poderá executar vários processos de servidor em cada instância. Isso pode reduzir o número de instâncias necessárias para atender à demanda dos jogadores.

Para obter mais informações:

- Sobre os tipos de instância, consulte [Tipos de EC2 instância da Amazon](#).
- Sobre a execução de vários processos por instância, consulte [Gerencie como Amazon GameLift Servers lança servidores de jogos](#).

Cotas de serviço

Para ver as cotas de serviço padrão para Amazon GameLift Server e as cotas atuais para você Conta da AWS, faça o seguinte:

- Para obter informações gerais sobre cotas de serviços para Amazon GameLift Servers, veja [Amazon GameLift Servers endpoints e cotas](#) no. Referência geral da AWS

- Para obter uma lista dos tipos de instância disponíveis por local para sua conta, abra a página [de cotas de serviço](#) do Amazon GameLift Servers console. Essa página também exibe o uso atual da sua conta para cada tipo de instância em cada local.
- Para obter uma lista das cotas atuais da sua conta para tipos de instância por região, execute o comando AWS Command Line Interface [describe-ec2-instance-limits](#)(AWS CLI). Esse comando retorna o número de instâncias ativas que você tem na sua região padrão (ou em outra região especificada).

Ao se preparar para lançar seu jogo, preencha um questionário de lançamento no [Amazon GameLift Servers console](#). A ferramenta Amazon GameLift Servers A equipe usa o questionário de lançamento para determinar as cotas e limites corretos para seu jogo.

Gerencie como Amazon GameLift Servers lança servidores de jogos

Você pode definir a configuração de tempo de execução de uma EC2 frota gerenciada para executar vários processos de servidor de jogos por instância. Isso usa os recursos de hospedagem com mais eficiência.

Como uma frota gerencia vários processos

Amazon GameLift Servers usa a configuração de tempo de execução de uma frota para determinar o tipo e o número de processos a serem executados em cada instância. Uma configuração de runtime contém pelo menos uma configuração de processo do servidor que representa um executável do servidor de jogos. É possível definir configurações adicionais do processo do servidor para executar outros tipos de processos relacionados ao seu jogo. Cada configuração do processo do servidor contém as seguintes informações:

- O nome do arquivo e o caminho de um executável na compilação do seu jogo.
- (Opcional) Parâmetros a serem passados para o processo na inicialização
- O número de processos a serem executados simultaneamente.

Quando uma instância na frota é ativada, ela inicia o conjunto de processos do servidor definido na configuração do runtime. Com vários processos, Amazon GameLift Servers escalona o lançamento de cada processo. Os processos do servidor têm uma vida útil limitada. Quando eles terminam, Amazon GameLift Servers inicia novos processos para manter o número e o tipo de processos do servidor definidos na configuração do tempo de execução.

É possível alterar a configuração de runtime a qualquer momento adicionando, alterando ou removendo as configurações de processo do servidor. Cada instância verifica regularmente se há atualizações na configuração de runtime da frota, para implementar as alterações. Veja como Amazon GameLift Servers adota alterações na configuração do tempo de execução:

1. A instância envia uma solicitação para Amazon GameLift Servers para a versão mais recente da configuração de tempo de execução.
2. A instância compara os processos ativos com a configuração de runtime mais recente e, em seguida, faz o seguinte:
 - Se a configuração atualizada do runtime remover um tipo de processo de servidor, os processos de servidor ativos desse tipo continuarão em execução até concluírem. A instância não substitui esses processos do servidor.
 - Se a configuração atualizada do runtime diminuir o número de processos simultâneos para um tipo de processo de servidor, os processos de servidor em excesso desse tipo continuarão em execução até terminarem. A instância não substitui esses processos excessivos do servidor.
 - Se a configuração de tempo de execução atualizada adicionar um novo tipo de processo do servidor ou aumentar os processos simultâneos de um tipo existente, a instância iniciará novos processos do servidor, até o Amazon GameLift Servers máximo. Nesse caso, a instância inicia novos processos de servidor quando os processos existentes são concluídos.

Otimize uma frota para vários processos

Para usar vários processos em uma frota, faça o seguinte:

- [Crie uma compilação](#) que contenha os executáveis do servidor de jogos que você deseja implantar em uma frota e, em seguida, carregue a compilação para Amazon GameLift Servers. Todos os servidores de jogos em uma compilação devem ser executados na mesma plataforma e usar o SDK do servidor para Amazon GameLift Servers.
- Crie uma configuração de runtime com uma ou mais configurações de processo de servidor e vários processos simultâneos.
- Integre clientes de jogos com a versão do AWS SDK 2016-08-04 ou posterior.

Para otimizar o desempenho da frota, é recomendável fazer o seguinte:

- Lide com cenários de desligamento do processo do servidor para que Amazon GameLift Servers pode reciclar processos de forma eficiente. Por exemplo:

- Adicione um procedimento de encerramento ao código do servidor de jogos que chama a API de servidor de `ProcessEnding()`.
- Implemente a função de retorno de chamada `OnProcessTerminate()` no código do servidor do jogo para lidar com solicitações de encerramento de Amazon GameLift Servers.
- Certifique-se de que Amazon GameLift Servers desliga e reinicia processos de servidor não íntegros. Relate o estado de saúde para Amazon GameLift Servers implementando a função de `OnHealthCheck()` retorno de chamada no código do servidor do jogo. Amazon GameLift Servers desliga automaticamente os processos do servidor que são relatados como não íntegros por três relatórios consecutivos. Se você não implementar `OnHealthCheck()`, então Amazon GameLift Servers pressupõe que um processo do servidor esteja íntegro, a menos que o processo falhe em responder a uma comunicação.

Escolha o número de processos por instância

Ao decidir em relação ao número de processos simultâneos a serem executados em uma instância, lembre-se do seguinte:

- Amazon GameLift Servers limita cada instância a um [número máximo de processos simultâneos](#). A soma de todos os processos simultâneos das configurações de processos do servidor de uma frota não pode exceder essa cota.
- Para manter níveis de desempenho aceitáveis, o tipo de EC2 instância da Amazon pode limitar o número de processos que podem ser executados simultaneamente. Teste diferentes configurações para o jogo a fim de encontrar o número certo de processos para o tipo de instância preferido.
- Amazon GameLift Servers não executa mais processos simultâneos do que o número total configurado. Isso significa que a transição da configuração de runtime anterior para a nova configuração pode ocorrer gradualmente.

Atualizar uma configuração de Amazon GameLift Servers frota

Use o Amazon GameLift Servers console ou a AWS CLI para atualizar as configurações da sua frota, alterar locais remotos ou excluir uma frota. Para frotas gerenciadas, não é possível alterar a compilação do servidor de jogos da frota ou o tipo de instância. Em vez disso, é necessário substituir a frota.

Fast Build Update Tool (somente para desenvolvimento)

Com EC2 frotas gerenciadas, para implantar uma atualização de compilação do servidor de jogos, você precisa carregar cada nova compilação Amazon GameLift Servers e criar uma nova frota para ela.

A Fast Build Update Tool permite que você ignore essas etapas durante o desenvolvimento, economizando tempo e permitindo uma iteração de desenvolvimento mais rápida. Com essa ferramenta, você pode atualizar rapidamente os arquivos de compilação de jogos em todos os computadores de uma frota existente. A ferramenta tem várias opções: você pode substituir uma versão inteira do jogo ou alterar 6 arquivos específicos e gerenciar como reiniciar os processos do servidor de jogos após as atualizações. Ela também permite atualizar computadores individuais em uma frota.

Para obter a Ferramenta de Atualização de Compilação Rápida e saber mais sobre como usá-la, visite o repositório do Amazon GameLift Servers Kit de Ferramentas da [Ferramenta de Atualização de Compilação Rápida](#) no Github.

Você pode atualizar atributos mutáveis da frota, configurações de porta e configurações de tempo de execução usando o Amazon GameLift Servers console ou a CLI AWS . Para alterar os limites de escala, consulte [Capacidade de frota de escala automática com Amazon GameLift Servers](#).

GameLift Servidores Amazon console

1. No [Amazon GameLift Serversconsole](#), no painel de navegação, escolha Fleets.
2. Selecione a frota que deseja atualizar. Uma frota deve ter o status ACTIVE antes de poder editá-la.
3. Na página de detalhes da frota, em qualquer uma das seções a seguir, escolha Editar.
 - Configurações da frota
 - Altere os atributos da frota, como Nome e Descrição.
 - Adicione ou remova grupos de métricas, que a Amazon CloudWatch usa para rastrear Amazon GameLift Servers métricas agregadas para várias frotas.
 - Atualize as configurações de Limite de criação de recursos.
 - Ative ou desative a proteção da sessão de jogo.
 - Configuração de runtime – Alterar qualquer uma das seguintes configurações de runtime e adicionar ou remover configurações de runtime.

- Altere o Caminho de inicialização do seu servidor de jogo.
 - Adicione, remova ou altere os Parâmetros opcionais de inicialização.
 - Altere o número de Processos simultâneos que seus servidores de jogos executam.
 - Ativação da sessão de jogo – Altere a forma como você deseja que os processos do servidor sejam executados e hospedem sessões de jogo atualizando o Máximo de ativações simultâneas de sessões de jogo e o Novo tempo limite de ativação.
 - EC2 configurações de porta — atualize os endereços IP e os intervalos de portas que permitem acesso de entrada à frota.
4. Depois, selecione Confirmar para salvar as alterações.

AWS CLI

Use os seguintes comandos da AWS CLI para atualizar uma frota:

- [update-fleet-attributes](#)
- [update-fleet-port-settings](#)
- [update-runtime-configuration](#)

Atualizar localizações da frota

Você pode adicionar ou remover os locais remotos de uma frota usando o Amazon GameLift Servers console ou a AWS CLI. Não é possível alterar a região de origem de uma frota.

GameLift Servidores Amazon console

1. No [Amazon GameLift Serversconsole](#), no painel de navegação, escolha Fleets.
2. Selecione a frota que deseja atualizar. Uma frota deve ter o status ACTIVE antes de poder editá-la.
3. Na página de detalhes da frota, escolha o guia Localizações para visualizar as localizações da frota.
4. Para adicionar novos locais remotos, escolha Adicionar e selecione os locais nos quais você deseja implantar instâncias. Essa lista não inclui instâncias em que o tipo de instância da frota não está disponível.

5. Com os novos locais selecionados, escolha Adicionar. Amazon GameLift Servers adiciona os novos locais à lista, com o status definido como NEW. Amazon GameLift Server em seguida, começa a provisionar uma instância em cada local adicionado e a prepara para hospedar sessões de jogo.
6. Para remover locais remotos existentes da frota, use as caixas de seleção para selecionar um ou mais locais listados.
7. Com uma ou mais frotas selecionadas, escolha Remover. Os locais removidos permanecem na lista, com o status definido como DELETING. Amazon GameLift Server em seguida, inicia o processo de encerramento da atividade no local removido. Se houver instâncias ativas que estejam hospedando sessões de jogo, Amazon GameLift Servers use o processo de encerramento do servidor de jogos para finalizar as sessões de jogo, encerrar servidores de jogos e desligar instâncias normalmente.

AWS CLI

Use os seguintes comandos da AWS CLI para atualizar os locais da frota:

- [create-fleet-locations](#)
- [delete-fleet-locations](#)

Excluir uma frota

É possível excluir uma frota quando não precisar mais dela. A exclusão de uma frota remove permanentemente todos os dados associados às sessões de jogos e de jogadores e dados de métricas coletados. Como alternativa, é possível reter a frota, desabilitar o ajuste de escala automático e escalar a frota manualmente para 0 instância.

Note

Se a frota tiver uma conexão de emparelhamento de VPC, primeiro solicite a autorização ligando. [CreateVpcPeeringAuthorization](#) Amazon GameLift Server exclui a conexão de emparelhamento de VPC durante a exclusão da frota.

Você pode usar o Amazon GameLift Servers console ou a ferramenta AWS CLI para excluir uma frota.

GameLift Servidores Amazon console

1. No [Amazon GameLift Servers console](#), no painel de navegação, escolha Fleets.
2. Escolha a frota que você deseja excluir. Você só pode excluir frotas no status ACTIVE ou ERROR.
3. Escolha Excluir.
4. Na caixa de diálogo Excluir frota, confirme a exclusão inserindo **delete**.
5. Escolha Excluir.

AWS CLI

Use o seguinte comando da AWS CLI para excluir uma frota:

- [delete-fleet](#)

Depurar problemas de frota do Amazon GameLift Servers

Este tópico fornece orientação sobre como resolver problemas com suas EC2 frotas Amazon GameLift Servers gerenciadas.

Depurar problemas de criação de frotas

Quando você cria uma EC2 frota gerenciada, o Amazon GameLift Servers serviço inicia um fluxo de trabalho que cria a frota, implanta EC2 instâncias com a compilação do servidor de jogos instalada e inicia os processos do servidor de jogos em cada instância. Para obter uma descrição detalhada, consulte [Como funciona a criação de Amazon GameLift Servers frotas](#). Uma frota não pode hospedar sessões de jogo e jogadores até atingir o status Ativo.

Você pode depurar problemas que impedem que as frotas se tornem ativas identificando a fase de criação da frota em que o problema ocorreu e analisando os eventos e logs de criação da frota. Se os logs não fornecerem informações úteis, é possível que o problema seja devido a um erro interno de serviço. Nesse caso, tente criar a frota novamente. Se o problema persistir, tente recarregar a compilação do jogo para resolver possíveis danos no arquivo. Também é possível entrar em contato com o suporte do Amazon GameLift Servers ou publicar uma pergunta no fórum.

Download e validação da compilação

Durante essa fase, Amazon GameLift Servers obtém a compilação do servidor de jogos carregada, extrai os arquivos e executa todos os scripts de instalação. Se a criação da frota falhar durante essas fases, analise os eventos e logs da frota para identificar o problema. As possíveis causas incluem:

- Amazon GameLift Servers não consegue obter o arquivo de compilação compactado (evento `FLEET_BINARY_DOWNLOAD_FAILED`). Verifique se o local de armazenamento da compilação pode ser acessado, se você está criando uma frota Região da AWS igual à da compilação e se Amazon GameLift Servers tem as permissões corretas para acessá-la.
- Amazon GameLift Servers não é possível extrair os arquivos de compilação (evento `FLEET_CREATION_EXTRACTING_BUILD`).
- Um script de instalação nos arquivos da compilação não foi concluído com êxito (evento `FLEET_CREATION_FAILED_INSTALLER`).

Compilar recursos de frota

Os problemas durante essa fase geralmente envolvem a alocação e a implantação de recursos da frota. As possíveis causas incluem:

- O tipo de instância solicitado não está disponível.
- O tipo de frota solicitado (spot ou sob demanda) não está disponível.

Ativar processos de servidores de jogos

Durante essa fase, Amazon GameLift Servers está tentando várias tarefas e testando elementos-chave, incluindo a viabilidade do servidor do jogo, as configurações de tempo de execução e a capacidade do servidor do jogo de se conectar ao Amazon GameLift Servers serviço usando o SDK do servidor.

Note

Nessa fase, você pode acessar remotamente uma instância de frota para investigar melhor os problemas. Consulte [Conecte-se remotamente a Amazon GameLift Servers instâncias de frota](#).

Os possíveis problemas incluem:

- Os processos do servidor não começam a ser executados. Isso sugere um problema com as configurações de tempo de execução da

frota (eventos `FLEET_VALIDATION_LAUNCH_PATH_NOT_FOUND` ou `FLEET_VALIDATION_EXECUTABLE_RUNTIME_FAILURE`). Verifique se você definiu corretamente o caminho de inicialização e os parâmetros de inicialização opcionais.

- Os processos de servidor começam a ser executados, mas a frota não consegue ser ativada. Se os processos do servidor forem iniciados e executados com êxito, mas a frota não passar para o status Ativo, uma causa provável é que o processo do servidor não esteja se comunicando com o Amazon GameLift Servers serviço. Verifique se o servidor do jogo está fazendo essas chamadas corretas do SDK do servidor (consulte [Inicializar o processo do servidor](#)):
 - Falha na inicialização do processo do servidor (evento `SERVER_PROCESS_SDK_INITIALIZATION_TIMEOUT`). O processo do servidor não está chamando `InitSdk()` com êxito.
 - O processo do servidor não notifica Amazon GameLift Servers quando está pronto para hospedar uma sessão de jogo (evento `SERVER_PROCESS_PROCESS_READY_TIMEOUT`). O processo do servidor foi inicializado, mas não chamou `ProcessReady()` a tempo.
- Uma solicitação de conexão de emparelhamento da VPC falhou. Para frotas que são criadas com uma conexão de emparelhamento de VPCs (consulte [Como configurar o emparelhamento de VPC com uma nova frota](#)), o emparelhamento de VPCs é feito durante essas fases de Ativação. Se um emparelhamento de VPCs falhar por qualquer motivo, haverá falha na mudança da nova frota para o status Ativo. Você pode acompanhar o sucesso ou a falha da solicitação de emparelhamento [describe-vpc-peering-connections](#) ligando. Certifique-se de verificar se existe uma autorização de emparelhamento de VPC válida ([describe-vpc-peering-authorizations](#)), já que as autorizações só são válidas por 24 horas.

Problemas de processo do servidor

Os processos do servidor começam, mas falham rapidamente ou apresentam integridade insatisfatória.

Outros problemas com a compilação do jogo, podendo acontecer quando se tenta executar muitos processos de servidor simultaneamente na instância. O número ideal de processos simultâneos depende do tipo de instância e dos requisitos de recursos do servidor de jogos. Tente reduzir o número de processos simultâneos, algo definido na configuração de tempo de execução da frota, para ver se o desempenho melhora. Você pode alterar a configuração de tempo de execução de uma frota usando o Amazon GameLift Servers console (editar as

configurações de alocação de capacidade da frota) ou chamando o AWS CLI comando [update-runtime-configuration](#).

Problemas de exclusão de frota

Uma frota não pode ser encerrada devido à contagem máxima de instâncias.

A mensagem de erro indica que a frota que está sendo excluída ainda tem instâncias ativas, o que não é permitido. Você deve primeiro reduzir uma frota para zero instâncias ativas. Isso é feito definindo manualmente a contagem de instâncias desejada da frota para "0" e aguardando até que a redução entre em vigor. Desligue a escalabilidade automática, pois ela neutralizará as configurações manuais.

As ações da VPC não estão autorizadas.

Esse problema se aplica somente às frotas para as quais você criou especificamente conexões de emparelhamento de VPC (consulte [Emparelhamento de VPC para Amazon GameLift Servers](#)). Esse cenário ocorre porque o processo de exclusão de uma frota também inclui a exclusão da VPC da frota e de qualquer conexão de emparelhamento de VPC. Primeiro, você deve obter uma autorização chamando a API de serviço for Amazon GameLift Servers [CreateVpcPeeringAuthorization\(\)](#) ou usar o comando AWS CLI. `create-vpc-peering-authorization` Quando tiver a autorização, você pode excluir a frota.

Amazon GameLift ServersRealtimeproblemas com a frota

Sessões de jogos zumbi: um jogo é iniciado e executado, mas nunca termina.

Você pode observar esses problemas como qualquer um dos seguintes cenários:

- Atualizações de script não são selecionadas por servidores Realtime da frota.
- A frota atinge a capacidade máxima rapidamente e não reduz quando as atividades dos jogadores (como novas solicitações de sessão de jogo) é reduzida.

Isso é quase certamente um resultado de não chamar `processEnding` com êxito em seu script Realtime. Embora a frota se torne ativa e as sessões de jogos sejam iniciadas, não há um método para interrompê-las. Como resultado, o servidor Realtime que está executando a sessão de jogo nunca é liberado para iniciar uma nova, e as novas sessões de jogos só podem ser iniciadas quando novos servidores Realtime forem rotacionados. Além disso, as atualizações no script Realtime não afetam as sessões de jogo que já estão em execução, apenas as novas.

Para impedir que isso aconteça, os scripts precisam fornecer um mecanismo para acionar uma chamada de `processEnding`. Conforme ilustrado no [Amazon GameLift Servers Realtime exemplo de script](#), uma maneira é programar um tempo limite de sessão ociosa em que, se nenhum jogador estiver conectado por um determinado período de tempo, o script encerrará a sessão do jogo atual.

No entanto, se você se encaixar nesse cenário, há duas alternativas para destravar seus servidores Realtime. O truque é acionar a reinicialização dos processos Realtime do servidor, ou das instâncias subjacentes da frota. Nesse caso, fecha Amazon GameLift Servers automaticamente as sessões do jogo para você. Assim que os servidores Realtime são liberados, eles podem iniciar novas sessões de jogos usando a versão mais recente do script Realtime.

Há dois métodos para fazer isso, dependendo de quão disseminado é o problema:

- Reduza toda a frota. Esse método é o mais simples de executar, mas tem um efeito amplo. Reduza a frota para zero instância, aguarde até que a frota seja totalmente reduzida e expanda-a novamente. Isso limpará todas as sessões de jogos existentes e permitirá que você comece de novo com o script Realtime atualizado mais recentemente.
- Acesse a instância remotamente e reinicie o processo. Esta será uma boa opção se você tiver apenas alguns processos para corrigir. Se você já estiver conectado à instância, como para logs finais ou depuração, esse poderá ser o método mais rápido. Consulte [Conecte-se remotamente a Amazon GameLift Servers instâncias de frota](#).

Se você optar por não incluir uma maneira de chamar `processEnding` em seu script Realtime, haverá algumas situações delicadas que poderão ocorrer, mesmo quando a frota estiver ativa e sessões de jogos estiverem iniciadas. Primeiro, uma sessão de jogo em execução não termina. Como resultado, o processo do servidor que está executando a sessão do jogo nunca é liberado para iniciar uma nova sessão de jogo. Em segundo lugar, o servidor Realtime não recebe nenhuma atualização de script.

Conecte-se remotamente a Amazon GameLift Servers instâncias de frota

Você pode se conectar a qualquer instância ativa Amazon GameLift Servers EC2 frotas gerenciadas. Motivos comuns para acessar remotamente uma instância incluem:

- Solucionar os problemas com a integração do servidor de jogos.
- Ajustar a configuração de tempo de execução e outras configurações específicas da frota.
- Obter atividade em tempo real do servidor de jogos, como monitoramento de logs.

- Executar ferramentas de benchmarking usando o tráfego real de jogadores.
- Investigar problemas específicos com uma sessão de jogo ou processo do servidor de jogos.

Ao se conectar a uma instância, considere estes possíveis problemas:

- É possível se conectar a qualquer instância em uma frota ativa. Geralmente, não é possível se conectar a frotas não ativas, como frotas que estão em processo de ativação ou em estado de erro. (Essas frotas podem ter disponibilidade limitada por um curto período.) Para obter ajuda com problemas de ativação da frota, consulte [Depurar problemas de frota do Amazon GameLift Servers](#).
- A conexão com uma instância ativa não afeta a atividade de hospedagem da instância. A instância continua a iniciar e interromper os processos do servidor com base na configuração do tempo de execução. Ela ativa e executa sessões de jogo. A instância pode ser encerrada em resposta a um evento de redução de escala vertical ou outro evento.
- Qualquer alteração que você fizer em arquivos ou configurações na instância podem impactar as sessões de jogo ativas e os jogadores conectados.

As instruções a seguir descrevem como se conectar remotamente a uma instância usando a interface de linha de comandos (CLI) da AWS . Você também pode fazer chamadas programáticas usando o AWS SDK, conforme documentado na referência da API [de serviço para Amazon GameLift Servers](#).

Coletar dados da instância

Para se conectar a um Amazon GameLift Servers instância de EC2 frota gerenciada, você precisa das seguintes informações:

- O ID da instância à qual você deseja se conectar. É possível utilizar o ID ou o ARN da instância.
- o SDK do servidor para Amazon GameLift Servers versão que está sendo usada na instância. O SDK do servidor é integrado à compilação do jogo que está sendo executada na instância.

As instruções a seguir descrevem como concluir essas tarefas usando a AWS CLI. É necessário saber o ID da frota da instância à qual você deseja se conectar.

1. Obtenha o nome da computação. Obtenha uma lista de todas os recursos de computação ativos na frota. Chame [list-compute](#) com um ID ou ARN de frota. Para uma frota de localização única,

especifique somente o identificador da frota. Para uma frota com vários locais, especifique o identificador da frota e um local. Com EC2 frotas gerenciadas, `list-compute` retorna uma lista de instâncias da frota e a propriedade `ComputeName` é o ID da instância. Localize o recurso de computação que deseja acessar.

Solicitação

```
aws gamelift list-compute \  
  --fleet-id "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa" \  
  --location "sa-east-1"
```

Resposta

```
{  
  "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":  
"b08444ki909kvqu6zpw3is24x5pyz4b6m05i3jbxvpk9craztu0lqrbbrbnnbkks.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. Localize a versão do SDK do servidor. Para obter essas informações, é necessário consultar a compilação que está implantada na frota. A versão do SDK do servidor é uma propriedade de compilação.
 - a. Ligue [describe-fleet-attributes](#) com um ID de frota ou ARN para obter o ID de construção e o ARN da frota.
 - b. Chame [describe-build](#) com o ID ou ARN de compilação para obter a versão do SDK do servidor de compilação.

Por exemplo:

Solicitação

```
aws gamelift describe-fleet-attributes /  
  --fleet-ids "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa"
```

Resposta

```
{  
  "FleetAttributes": [  
    {  
      "FleetId": "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",  
      "ComputeType": "EC2",  
      "BuildId": "build-3333cccc-44dd-55ee-66ff-00001111aa22",  
      . . .  
    }  
  ]  
}
```

Solicitação

```
aws gamelift describe-build /  
  --build-id "build-3333cccc-44dd-55ee-66ff-00001111aa22"
```

Resposta

```
"Build": {  
  "BuildId": "build-1111aaaa-22bb-33cc-44dd-5555eeee66ff",  
  "Name": "My_Game_Server_Build_One",  
  "OperatingSystem": "AMAZON_LINUX_2023",  
  
  "ServerSdkVersion": "5.1.1",  
  . . .  
}
```

Conectar-se a uma instância (SDK 5 do servidor)

Se a instância à qual você deseja se conectar estiver executando uma compilação de jogo com o SDK do servidor versão 5.x, conecte-se à instância usando o Amazon EC2 Systems Manager (SSM). Você pode acessar instâncias remotas que estejam executando o Windows ou o Linux.

Antes de começar:

Conclua as etapas de configuração do SSM e instale o plug-in do SSM em sua máquina local. Para obter mais informações, consulte [Configurar o SSM](#) e [instalar o plug-in do Session Manager para a AWS CLI no](#) Guia do usuário do Amazon Systems EC2 Manager.

1. Solicite credenciais de acesso para a instância. Ligue [get-compute-access](#) com o ID da frota e o nome de computação da instância à qual você quer se conectar. Amazon GameLift Servers retorna um conjunto de credenciais temporárias para acessar a instância. Por exemplo:

Solicitação

```
aws gamelift get-compute-access \  
--compute-name i-11111111a222b333c \  
--fleet-id fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa \  
--region us-west-2
```

Resposta

```
{  
  "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. Exporte as credenciais de acesso (opcional). Você pode exportar as credenciais para variáveis de ambiente e usá-las para configurar a AWS CLI para o usuário padrão. Para obter mais

detalhes, consulte [Variáveis de ambiente para configurar a AWS CLI](#) no Guia do AWS Command Line Interface usuário.

```
export AWS_ACCESS_KEY_ID=ASIAIOSFODNN7EXAMPLE
export AWS_SECRET_ACCESS_KEY=wJa1rXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY
export AWS_SESSION_TOKEN=AQoDYXdzEJr...<remainder of session token>
```

3. Conecte-se à instância da frota. Inicie uma sessão de SSM com a instância à qual você deseja se conectar. Inclua a AWS região ou o local da instância. Para obter mais informações, incluindo como configurar o SSM e o plug-in SSM, consulte [Iniciando uma sessão \(AWS CLI\)](#) no Guia do usuário do Amazon EC2 Systems Manager.

A solicitação de início de sessão usará automaticamente as credenciais que você adquiriu na Etapa 1.

```
aws ssm start-session \
--target i-11111111a222b333c \
--region us-west-2 \
```

Note

Se você receber um erro de acesso negado, talvez tenha uma variável de `AWS_PROFILE` ambiente definida como um AWS perfil, o que faz com que a AWS CLI use as credenciais erradas para acesso remoto. Para resolver, cancele temporariamente a variável de ambiente `AWS_PROFILE`. Como alternativa, você pode criar um AWS perfil personalizado para suas credenciais de acesso remoto e adicionar o parâmetro da linha de `--profile` comando à sua `start-session` solicitação.

Conectar-se a uma instância (SDK 4.x ou anterior do servidor)

Se a instância à qual você deseja se conectar estiver executando uma compilação de jogo com o SDK versão 4 ou anterior do servidor, use as instruções a seguir. Você pode se conectar a instâncias que estejam executando o Windows ou o Linux. Conecte-se a uma instância do Windows usando um cliente de protocolo de área de trabalho remota (RDP). Conecte-se a uma instância do Linux usando um cliente SSH.

1. Solicite credenciais de acesso para a instância. Quando você tiver um ID de instância, use o comando [get-instance-access](#) para solicitar credenciais de acesso. Se for bem-sucedido,

Amazon GameLift Servers retorna o sistema operacional, o endereço IP e um conjunto de credenciais da instância (nome de usuário e chave secreta). O formato de credenciais depende do sistema operacional da instância. Use as instruções a seguir para recuperar credenciais de RDP ou SSH.

- Para instâncias do Windows – Para se conectar a uma instância do Windows, o RDP requer um nome de usuário e uma senha. A solicitação `get-instance-access` retorna esses valores como strings simples, de maneira que você possa usar os valores retornados como estão. Exemplo de credenciais:

```
"Credentials": {
  "Secret": "aA1bBB2cCCd3EEE",
  "UserName": "gl-user-remote"
}
```

- Para instâncias do Linux – Para se conectar a uma instância do Linux, o SSH requer um nome de usuário e uma chave privada. Amazon GameLift Servers emite chaves privadas RSA e as retorna como uma única string, com o caractere de nova linha (`\n`) indicando quebras de linha. Para tornar a chave privada utilizável, siga estas etapas: (1) converter a string em um arquivo `.pem` e (2) definir permissões para o novo arquivo. Exemplo de credenciais retornadas:

```
"Credentials": {
  "Secret": "-----BEGIN RSA PRIVATE KEY-----
nEXAMPLEKEYKCAQEAY7WZhaDsrA1W3mRlQtvhwyORRX8gnxgDAfRt/gx42kWXsT4rXE/b5CpSgie/
\nvBoU7jLxx92pNHoFnByP+Dc21eyyz6CvjTmWA0JwfWiW5/akH7i05dSrvC7dQkW2duV5QuUdE0QW
\nZ/aNxMniGQE6XAgfwlnXVBwrerrQo+ZWQeqiUwwMkuEbLeJFLhMCvYURpUMSC1oehm449i1x9X1F
\nG50TCFe0zf18dqqCP6GzbPaIjiU19xX/az0R9V+tpU0zEL+wmXnZt3/nHPQ5xvD20JH67km6SuPW
\nnoPzev/D8V+x4+bHthfSjR9Y7DvQFjfbVwHXigBdtZcU2/wei8D/HYwIDAQABaoIBAGZ1kaEvnrrqu
\n/uler7vgIn5m7lN5LKw4hJLAIW6tUT/fzvtCHK0SkbQCQXuriHmQ2MQyJX/0kn2NfjLV/
ufGxbL1\nmb5qwmGUNepJaZD6QSSs3kICLwUYUiGfc0uiSbmJoap/
GTLU0W5Mfcv36PaBUNy5p53V6G7hXb2\nbahyWyJNfjLe4M86yd2YK3V2CmK+X/
B0sShnJ36+hjrXPPWmV3N9zEmCdJjA+K15DYmhm/
tJWSD9\n81oGk9TopEp7CkIfatEATyyZiVqoRq6k64iuM9JkA30zdXzMqexXVJ1TLZVEH0E7bh1Y9d801ozR
\noQs/FiZNAx2iijCWyv0lpjE73+kCgYEA9mZtyhkHkFDpwrSM1APaL8oNAbbjwEy7Z5Mqfq1
+1Ip1\nYkriL0DbLXlvRAH+yHPRit2hH0jtUNZh4Axv+cpq09qbUI3+43eEy24B7G/Uh
+GTfbjsXs0xQx/x\np9otyVwc7hsQ5TA5PZb
+mvkJ50BEKzet9XCkKw0NBVELGhnEPe7cCgYEA06Vgov6YHleHui9kHuws
\nayav0elc5zkxjF9nfHFJRry21R1trw2Vdpn+9g481URrpzWV0Eihvm+xTtmaZ1Sp//lkq75XDwnU
\nWA8gkn603QE3fq2yN98BURsAKdJfJ5RL1HvGQvTe10HLYYXpJnEkHv+Unl2ajLivWUt5pbBrKbUC
\nngYBjb0+0Zk0sCcpZ29sbzjYjpIddErySIyRX5gV2uNQwAjLdp9PfN295yQ+BxMBXiIycWVQiw0bH
-----"
```

```

\noMo7yykABY70zd5wQewBQ4AdSlWSX4nGDtsiFxiWiI5sKuAAe0CbTosy1s8w8fxoJ5Tz1sdoxNeGs
\nArq6Wv/G16zQuAE9zK9vvwKBgF+09VI/1wJBirsDGz9whVwFFPrTkJNVJZzYt69qezx1sjgFKshy
\nWBhd4xHZtmCqpBP1AymEjr/T01bxyARmXMnIOWIAnNXMGB4KGSyl1mzSVAoQ+fqR+cJ3d0dyP11j
\njjb0Ed/NY8frlNDxAVHE8BSkdsx2f6ELEyBKJSRr9snRAoGAMrTwYneXzvTskF/S5Fyu0i0egLda
\nNWUH38v/nDCgEpIXD5Hn3qAEcju1IjmbwlvtW+nY2jVhv7UGd8MjwUTNGItdb6nsYqM2asrnF3qS
\nVRkAKKKYeGjKpUfVTrW0YFjXkfcrR/V+QFL50ndHAKJXjW7a4ejJLncTzmZSpYzwApc=\n-----END
RSA PRIVATE KEY-----",
    "UserName": "gl-user-remote"
}

```

Ao usar a AWS CLI, você pode gerar automaticamente um `.pem` arquivo incluindo os parâmetros `--query` e `--output` em sua solicitação. `get-instance-access`

Para definir permissões no arquivo `.pem`, execute o comando a seguir:

```
$ chmod 400 MyPrivateKey.pem
```

2. Abra uma porta para a conexão remota. Você pode acessar instâncias em Amazon GameLift Servers frotas através de qualquer porto autorizado na configuração da frota. Você pode visualizar as configurações de porta da frota usando o comando [describe-fleet-port-settings](#).

Como melhor prática, recomendamos só abrir portas para acesso remoto quando você precisar delas e fechá-las quando terminar. Não é possível atualizar as configurações da porta depois de criar uma frota e antes que ela esteja ativa. Se ocorrer algum problema, recrie a frota com as portas configuradas como abertas.

Use o comando [update-fleet-port-settings](#) para adicionar uma configuração de porta para a conexão remota (como 22 para SSH ou 3389 para RDP). Para o valor do intervalo de IPs, especifique os endereços IP dos dispositivos que você pretende usar para se conectar (convertidos em formato CIDR). Exemplo: .

```

$ 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"

```

O exemplo a seguir abre a porta 3389 em uma frota do 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. Abra um cliente de conexão remota. Use Área de Trabalho Remota para Windows ou SSH para instâncias do Linux. Conecte-se à instância usando o endereço IP, a configuração de porta e as credenciais de acesso.

Exemplo de SSH:

```
ssh -i MyPrivateKey.pem gl-user-remote@192.0.2.0
```

Visualizar arquivos em instâncias remotas

Quando conectado a uma instância remotamente, você tem acesso administrativo e de usuário total. Isso significa que você também pode causar erros e falhas na hospedagem de jogos. Se a instância estiver hospedando jogos com jogadores ativos, há o risco de encerrar as sessões de jogo e desconectar jogadores ou interromper processos de desligamento de jogos e causar erros em dados e logs de jogos salvos.

Procure esses recursos em uma instância de hospedagem:

- Arquivos de compilação do jogo. Esses arquivos são a versão do jogo para a qual você fez o upload. Amazon GameLift Servers. Eles incluem um ou mais executáveis, ativos e dependências do servidor de jogos. Os arquivos de compilação do jogo estão em um diretório raiz chamado game:
 - No Windows: c:\game
 - No Linux: /local/game
- Arquivos de log do jogo. Os arquivos de log que o servidor de jogo gera ficam no diretório raiz game, no caminho de diretório designado para logs.
- Amazon GameLift Servers recursos de hospedagem. O diretório raiz Whitewater contém arquivos usados pelo Amazon GameLift Servers serviço para gerenciar a atividade de hospedagem de jogos. Não modifique esses arquivos por nenhum motivo.
- Configuração de runtime. Não acesse configurações de tempo de execução para instâncias individuais. Para fazer alterações em uma propriedade de configuração de tempo de execução, atualize a configuração de tempo de execução da frota (consulte a operação do AWS SDK [UpdateRuntimeConfiguration](#) ou a AWS CLI [update-runtime-configuration](#)).

- Dados da frota. Um arquivo JSON contém informações sobre a frota à qual a instância pertence, para uso pelos processos do servidor em execução na instância. O arquivo JSON está no seguinte local:
 - No Windows: `C:\GameMetadata\gamelift-metadata.json`
 - No Linux: `/local/gamemetadata/gamelift-metadata.json`
- Certificados TLS. Se a instância estiver em uma frota com geração de certificado TLS habilitada, você encontrará os arquivos de certificado, incluindo o certificado, a cadeia de certificados, a chave privada e o certificado raiz no seguinte local:
 - No Windows: `c:\GameMetadata\Certificates`
 - No Linux: `/local/gamemetadata/certificates/`

Emparelhamento de VPC para Amazon GameLift Servers

Este tópico fornece orientação sobre como configurar uma conexão de emparelhamento de VPC entre seus Amazon GameLift Servers-servidores de jogos hospedados e seus outros servidores não-Amazon GameLift Servers recursos. Use as conexões de emparelhamento da Amazon Virtual Private Cloud (VPC) para permitir que seus servidores de jogos se comuniquem direta e privadamente com seus outros AWS recursos, como um serviço web ou um repositório. Você pode estabelecer o emparelhamento de VPC com qualquer recurso executado AWS e gerenciado por uma AWS conta à qual você tenha acesso.

Note

O emparelhamento de VPCs é um recurso avançado. Para saber mais sobre as opções preferidas para permitir que seus servidores de jogos se comuniquem direta e privadamente com seus outros AWS recursos, consulte [Comunique-se com outros AWS recursos de suas frotas](#).

Se você já está familiarizado com o peering da Amazon VPCs e do VPC, entenda que configurar o peering com Amazon GameLift Servers os servidores de jogos são um pouco diferentes. Você não tem acesso à VPC que contém seus servidores de jogos. Ela é controlada pelo Amazon GameLift Servers serviço — portanto, você não pode solicitar diretamente o emparelhamento de VPC para ele. Em vez disso, primeiro você pré-autoriza a VPC com seu não-Amazon GameLift Servers recursos para aceitar uma solicitação de emparelhamento do Amazon GameLift Servers serviço. Então você aciona Amazon GameLift Servers para solicitar o emparelhamento de VPC que você acabou de

autorizar. Amazon GameLift Servers executa as tarefas de criar a conexão de emparelhamento, configurar as tabelas de rotas e configurar a conexão.

Como configurar o emparelhamento de VPC para uma frota existente

1. Obtenha IDs e credenciais da AWS conta.

Você precisa de um ID e credenciais de login para as seguintes contas. AWS Você pode encontrar AWS uma conta IDs fazendo login [AWS Management Console](#) e visualizando as configurações da sua conta. Para obter as credenciais, acesse o console do IAM.

- AWS conta que você usa para gerenciar seu Amazon GameLift Servers servidores de jogos.
- AWS conta que você usa para gerenciar sua conta nãoAmazon GameLift Servers recursos.

Se você estiver usando a mesma conta para Amazon GameLift Servers e não-A Amazon GameLift Servers recursos, você precisa de ID e credenciais somente para essa conta.

2. Obtenha identificadores para cada VPC.

Obtenha as seguintes informações para que os dois VPCs sejam examinados:

- VPC para o seu Amazon GameLift Servers servidores de jogos — Este é o seu Amazon GameLift Servers ID da frota. Seus servidores de jogos estão implantados em Amazon GameLift Servers em uma frota de EC2 instâncias. Uma frota é automaticamente colocada em sua própria VPC, que é gerenciada pelo Amazon GameLift Servers serviço. Você não tem acesso direto à VPC, e isso é identificado pelo ID da frota.
- VPC para quem não éAmazon GameLift Servers AWS recursos — você pode estabelecer um emparelhamento de VPC com qualquer recurso executado AWS e gerenciado por uma AWS conta à qual você tenha acesso. Se você ainda não criou uma VPC para esses recursos, consulte [Conceitos básicos da Amazon VPC](#). Depois de criar uma VPC, você pode encontrar a ID da VPC fazendo login na [AWS Management Console](#) Amazon VPC e visualizando sua. VPCs

Note

Ao configurar um peering, ambos VPCs devem existir na mesma região. O VPC para o seu Amazon GameLift Servers os servidores de jogos da frota estão na mesma região da frota.

3. Autorize um emparelhamento de VPC.

Nesta etapa, você está pré-autorizando uma solicitação futura de Amazon GameLift Servers para emparelhar a VPC com seus servidores de jogos com sua VPC para nãoAmazon GameLift Servers recursos. Esta ação atualiza o grupo de segurança da sua VPC.

Para autorizar o emparelhamento de VPC, chame a [CreateVpcPeeringAuthorizationAPI de serviço \(\)](#) ou use o AWS comando CLI. `create-vpc-peering-authorization` Faça essa ligação usando a conta que gerencia sua conta não-A Amazon GameLift Servers recursos. Identifique as seguintes informações:

- ID de VPC de mesmo nível — Isso é para o VPC com seu nãoAmazon GameLift Servers recursos.
- Amazon GameLift Servers AWS ID da conta — Esta é a conta que você usa para gerenciar sua Amazon GameLift Servers frota.

Após autorizar um emparelhamento da VPC, a autorização permanecerá válida por 24 horas, a menos que seja revogada. É possível gerenciar suas autorizações de emparelhamento de VPCs usando as seguintes operações:

- [DescribeVpcPeeringAuthorizations\(\)](#) (AWS CLI `describe-vpc-peering-authorizations`).
- [DeleteVpcPeeringAuthorization\(\)](#) (AWS CLI `delete-vpc-peering-authorization`).

4. Solicite uma conexão de emparelhamento.

Com uma autorização válida, você pode solicitar que Amazon GameLift Servers estabeleça uma conexão de peering.

Para solicitar um emparelhamento de VPC, chame a API de serviço [CreateVpcPeeringConnection\(\) ou use](#) o comando CLI AWS. `create-vpc-peering-connection` Faça essa chamada usando a conta que gerencia seu Amazon GameLift Servers servidores de jogos. Use as informações a seguir para identificar os dois VPCs que você deseja comparar:

- ID de VPC de mesmo nível AWS e ID da conta — Esta é a VPC para quem não éAmazon GameLift Servers recursos e a conta que você usa para gerenciá-los. O ID da VPC deve corresponder ao ID em uma autorização de emparelhamento válida.
- ID da frota — Isso identifica a VPC para sua Amazon GameLift Servers servidores de jogos.

5. Acompanhe o status da conexão de emparelhamento.

Solicitar uma conexão de emparelhamento da VPC é uma operação assíncrona. Para rastrear o status de uma solicitação de emparelhamento e processar os casos de sucesso ou falha, use uma das seguintes opções:

- Sonde continuamente com `DescribeVpcPeeringConnections()`. Essa operação recupera o registro de conexão de emparelhamento da VPC, incluindo o status da solicitação. Se uma conexão emparelhada for criada com êxito, o registro de conexão também conterá um bloco CIDR de endereços IP privados que é atribuído à VPC.
- Gerencie eventos de frota associados a conexões de emparelhamento de VPC com [DescribeFleetEvents\(\)](#), incluindo eventos de sucesso e falha.

Uma vez estabelecida a conexão de emparelhamento, é possível gerenciá-la usando as seguintes operações:

- [DescribeVpcPeeringConnections\(\)](#) (AWS CLI `describe-vpc-peering-connections`).
- [DeleteVpcPeeringConnection\(\)](#) (AWS CLI `delete-vpc-peering-connection`).

Como configurar o emparelhamento de VPC com uma nova frota

Você pode criar um novo Amazon GameLift Servers crie uma frota e solicite uma conexão de emparelhamento de VPC ao mesmo tempo.

1. Obtenha IDs e credenciais da AWS conta.

Você precisa de um ID e credenciais de login para as duas contas a seguir. AWS Você pode encontrar AWS uma conta IDs fazendo login [AWS Management Console](#) visualizando as configurações da sua conta. Para obter as credenciais, acesse o console do IAM.

- AWS conta que você usa para gerenciar seu Amazon GameLift Servers servidores de jogos.
- AWS conta que você usa para gerenciar sua conta nãoAmazon GameLift Servers recursos.

Se você estiver usando a mesma conta para Amazon GameLift Servers e não-Amazon GameLift Servers recursos, você precisa de ID e credenciais somente para essa conta.

2. Obtenha o ID da VPC para quem não é Amazon GameLift Servers recursos.

Se você ainda não criou uma VPC para esses recursos, faça isso agora (consulte [Conceitos básicos da Amazon VPC](#)). Certifique-se de criar a nova VPC na mesma região em que planeja criar sua nova frota. Se você não... Amazon GameLift Servers os recursos são gerenciados em uma AWS conta ou usuário/grupo de usuários diferente daquele que você usa com Amazon GameLift Servers, você precisará usar essas credenciais da conta ao solicitar autorização na próxima etapa.

Depois de criar uma VPC, você pode localizar a ID da VPC no console da Amazon VPC visualizando sua. VPCs

3. Autorize um emparelhamento de VPC com um não-Amazon GameLift Servers recursos.

Quando Amazon GameLift Servers cria a nova frota e uma VPC correspondente, além de enviar uma solicitação para emparelhar a VPC para quem não é Amazon GameLift Servers recursos. Você precisa pré-autorizar essa solicitação. Esta etapa atualiza o grupo de segurança da sua VPC.

Usando as credenciais da conta que gerenciam suas informações não Amazon GameLift Servers recursos, chame a API de serviço [CreateVpcPeeringAuthorization\(\)](#) ou use o AWS comando CLI. `create-vpc-peering-authorization` Identifique as seguintes informações:

- ID de VPC de mesmo nível — ID da VPC com sua pessoa que não é Amazon GameLift Servers recursos.
- Amazon GameLift Servers AWS ID da conta — ID da conta que você usa para gerenciar sua Amazon GameLift Servers frota.

Após autorizar um emparelhamento da VPC, a autorização permanecerá válida por 24 horas, a menos que seja revogada. É possível gerenciar suas autorizações de emparelhamento de VPCs usando as seguintes operações:

- [DescribeVpcPeeringAuthorizations\(\)](#) (AWS CLI `describe-vpc-peering-authorizations`).
- [DeleteVpcPeeringAuthorization\(\)](#) (AWS CLI `delete-vpc-peering-authorization`).

4. Siga as instruções para [criar uma nova frota usando a AWS CLI](#). Inclua os parâmetros adicionais a seguir:

- `peer-vpc-aws-account-id` — ID da conta que você usa para gerenciar a VPC com pessoas que não são Amazon GameLift Servers recursos.
- `peer-vpc-id` — ID da VPC com seu não-A Amazon GameLift Servers conta.

Uma chamada bem-sucedida para [create-fleet](#) com os parâmetros de emparelhamento de VPC gera uma nova frota e uma nova solicitação de emparelhamento de VPC. O status da frota é definido para Novo, e o processo de ativação dela é iniciado. O status da solicitação de conexão de emparelhamento é definido como `initiating-request`. Você pode acompanhar o sucesso ou a falha da solicitação de emparelhamento [describe-vpc-peering-connections](#) ligando.

Ao solicitar uma nova frota e uma conexão de emparelhamento da VPC, ambas as ações serão bem-sucedidas ou falharão. Se uma frota falhar durante o processo de criação, a conexão de emparelhamento da VPC não será estabelecida. Da mesma forma, se uma conexão de emparelhamento da VPC falhar por qualquer motivo, a nova frota falhará ao mudar do status Ativando para Ativo.

Note

A nova conexão de emparelhamento da VPC não será concluída até que a frota esteja pronta para se tornar ativa. Isso significa que a conexão não está disponível e não pode ser usada durante o processo de instalação de compilação do servidor de jogos.

O exemplo a seguir cria uma nova frota e uma conexão de emparelhamento entre uma VPC pré-estabelecida e a VPC para a nova frota. A VPC preestabelecida é identificada exclusivamente pela combinação de sua VPC não estabelecida Amazon GameLift Servers AWS ID da conta e ID da 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"

```

Versão copiável:

```

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"

```

Solucionar problemas de emparelhamento de VPC

Se você estiver tendo problemas para estabelecer uma conexão de emparelhamento de VPC para seu Amazon GameLift Servers servidores de jogos, considere estas causas básicas comuns:

- Uma autorização para a conexão solicitada não foi encontrada:
 - Verifique o status de uma autorização de VPC para pessoas que não são Amazon GameLift Servers VPC. Pode não existir ou pode ter expirado.
 - Verifique as regiões das duas VPCs que você está tentando examinar. Se não estiverem na mesma região, elas não podem ser emparelhadas.
- Os blocos CIDR (consulte [Configurações de conexão de emparelhamento de VPC inválidas](#)) de seus dois estão sobrepostos. VPCs Os blocos IPv4 CIDR atribuídos ao peering VPCs não podem se sobrepor. O bloco CIDR da VPC para o seu Amazon GameLift Servers a frota é atribuída

automaticamente e não pode ser alterada, então você precisará alterar o bloco CIDR da VPC para sua frota que não seja Amazon GameLift Servers recursos. Para resolver esse problema:

- Procure este bloco CIDR para o seu Amazon GameLift Servers frota ligando `DescribeVpcPeeringConnections()`.
- Acesse o console da Amazon VPC, encontre a VPC para quem não é Amazon GameLift Servers recursos e altere o bloco CIDR para que eles não se sobreponham.
- A nova frota não foi ativada (ao solicitar o emparelhamento de VPC com uma nova frota). Se a nova frota não conseguir avançar para o status Ativa não haverá VPC com a qual emparelhar, portanto, a conexão de emparelhamento não poderá ser bem-sucedida.

Amazon GameLift Servers frotas de contêineres gerenciadas

Amazon GameLift Servers frotas de contêineres gerenciados oferecem uma plataforma baseada em nuvem para hospedar seu software de servidor de jogos em contêineres. Com uma frota de contêineres, você obtém a flexibilidade, a segurança e a confiabilidade dos Nuvem AWS recursos, que são ainda mais otimizados para hospedagem de jogos multijogador. O Amazon GameLift Servers serviço fornece ferramentas robustas de gerenciamento de host.

 Acelere a integração com essas ferramentas para contêineres gerenciados:

- O [kit inicial de contêineres](#) simplifica a integração e a configuração da frota. Ele adiciona recursos essenciais de gerenciamento de sessões de jogos ao seu servidor de jogos e usa modelos pré-configurados para criar uma frota de contêineres e um pipeline de implantação automatizado para seu servidor de jogos. Após a implantação, use o Amazon GameLift Servers console e as ferramentas de API para monitorar o desempenho da frota, gerenciar sessões de jogo e analisar métricas.
- Para desenvolvedores do Unreal Engine ou Unity, use os [Amazon GameLift Servers plug-ins e o servidor](#) do mecanismo de jogo SDKs para integrar seu servidor de jogos e criar uma frota de contêineres de dentro do ambiente de desenvolvimento do seu mecanismo de jogo. Os fluxos de trabalho guiados do plug-in ajudam você a criar uma solução rápida e simples com hospedagem baseada em nuvem usando contêineres gerenciados. Você pode aproveitar essa base para criar uma solução de hospedagem personalizada para seu jogo.

Uma frota gerenciada de contêineres é composta por um conjunto de computadores virtuais que Amazon GameLift Servers possuem e operam em seu nome e com base em suas opções de configuração. As computações são instâncias do Amazon Elastic Compute Cloud EC2 (Amazon) com o sistema operacional Amazon GameLift Servers Linux. As instâncias estão localizadas fisicamente em Regiões da AWS nossas Zonas Locais. Ao criar uma frota de contêineres, você escolhe um tipo de EC2 instância para seus computadores com base na capacidade de computação, memória, armazenamento, recursos de rede e outros fatores.

Para uma frota de contêineres gerenciada, você armazena imagens de contêineres baseadas em Linux em um repositório do Amazon Elastic Container Registry (Amazon ECR) e cria uma definição de grupo de contêineres para descrever sua arquitetura de contêineres. Quando você cria uma frota, o Amazon GameLift Servers serviço usa a definição do grupo de contêineres para implantar suas imagens de contêiner nas instâncias da frota. À medida que os contêineres iniciam os processos do servidor de jogos, cada processo estabelece uma conexão com o Amazon GameLift Servers serviço e relata a prontidão para hospedar uma sessão de jogo.

Além da implantação da frota, Amazon GameLift Servers realiza as seguintes tarefas de gerenciamento do host para que você não precise:

- Rastreia o status de todos os contêineres da frota e substitui os obsoletos ou insalubres.
- Lida com a autenticação para comunicação entre os processos do servidor e o Amazon GameLift Servers serviço.
- Oferece ferramentas de ajuste de escala automático que ajustam dinamicamente a capacidade da frota para atender à demanda dos jogadores.
- Relata métricas de desempenho para as EC2 instâncias, contêineres e processos do servidor da frota.

Veja estes tópicos sobre como configurar e manter frotas de contêineres gerenciadas:

- [Roteiro de desenvolvimento para hospedagem com contêineres Amazon GameLift Servers gerenciados](#)
- [Crie um Amazon GameLift Servers frota de contêineres gerenciada](#)
- [Personalize uma frota de Amazon GameLift Servers contêineres](#)
- [Escalando a capacidade de hospedagem de jogos com Amazon GameLift Servers](#)
- [Atualizar um Amazon GameLift Servers frota de contêineres gerenciada](#)

Crie um Amazon GameLift Servers frota de contêineres gerenciada

Crie um Amazon GameLift Servers frota de contêineres gerenciada para implantar e hospedar seu servidor de jogos em contêineres na AWS nuvem. Ao criar uma frota de contêineres, você pode especificar definições de grupos de contêineres com configurações, incluindo uma ou mais imagens de contêineres, incluindo uma imagem com seu servidor de jogo. Você também pode optar por criar uma frota de contêineres vazia e adicionar ou atualizar as definições do grupo de contêineres da frota posteriormente.

Você pode usar o [Amazon GameLift Servers console](#) ou o AWS Command Line Interface (AWS CLI) para criar uma frota de contêineres.

Depois de criar uma nova frota de contêineres, o status da frota passa por vários estágios como Amazon GameLift Servers implanta seus grupos de contêineres em cada instância da frota e inicia os servidores do jogo. Quando a frota atingir o status `ACTIVE`, ela estará pronta para sediar sessões de jogo. Se você criar uma frota sem uma definição de grupo de contêineres, a frota não alcançará o status ativo. Para obter ajuda com problemas de criação de frotas, consulte [Depurar problemas de frota do Amazon GameLift Servers](#).

Console

No [Amazon GameLift Servers console](#), selecione Região da AWS onde você deseja criar a frota. As definições do grupo de contêineres devem estar na mesma região em que você deseja criar a frota.

Abra a barra de navegação esquerda do console e escolha Contêineres gerenciados: frotas. Na página Frotas, escolha Criar frota de contêineres.

Etapa 1: Definir detalhes da frota de contêineres gerenciados

1. Na seção Detalhes da frota de contêineres, insira uma descrição da frota.
2. Especifique uma função do IAM para a frota. Essa função tem permissões que Amazon GameLift Servers deve ter que gerenciar a frota de contêineres em seu nome. Para obter ajuda na criação da função de serviço necessária, consulte [Configure uma função de serviço do IAM para Amazon GameLift Servers](#).
3. Escolha uma opção de configuração de log. A CloudWatch opção é selecionada por padrão. Forneça as informações necessárias com base na opção selecionada.
4. Adicione grupos de contêineres à frota. Esta é uma etapa opcional. Você pode optar por criar uma frota sem um grupo de contêineres com um plano para adicioná-los posteriormente.

Uma frota sem grupos de contêineres não implantará nenhuma instância de frota e ainda não poderá hospedar nenhum jogo, mas o recurso da frota foi criado.

- Selecione uma definição de grupo de contêineres do servidor de jogos. Opcionalmente, especifique a versão da definição que você deseja implantar. Se você não especificar o número da versão, Amazon GameLift Servers usa automaticamente a versão mais recente.
 - Opcionalmente, adicione uma definição e uma versão do grupo de contêineres por instância. Se você não especificar o número da versão, Amazon GameLift Servers usa automaticamente a versão mais recente.
5. Em Detalhes adicionais, você pode definir algumas personalizações opcionais. Nenhuma dessas configurações é necessária para criar a frota de contêineres.

Etapa 2: definir detalhes da instância

1. Em Implantação de instâncias, selecione um ou mais locais remotos para implantar instâncias. A região de origem é selecionada automaticamente (essa é a região na qual você está criando a frota). Caso selecione locais adicionais, as instâncias da frota também serão implantadas nesses locais.

Important

Para usar regiões que não estão habilitadas por padrão, ative-as em seu Conta da AWS.

- As frotas com regiões não habilitadas que você criou antes de 28 de fevereiro de 2022 não são afetadas.
- Para criar novas frotas com vários locais ou atualizar frotas com vários locais existentes, primeiro ative todas as regiões que você escolher usar.

Para obter mais informações sobre regiões que não estão habilitadas por padrão e como habilitá-las, consulte [Gerenciando Regiões da AWS](#) no Referência geral da AWS.

2. Selecione uma configuração de instância para a frota. O console calcula automaticamente a vCPU e a memória mínimas necessárias (com base nos limites totais definidos para cada grupo de contêineres). Ele filtra a lista completa dos tipos de instância disponíveis com

base nos requisitos de recursos e nos locais que você inseriu. Você pode adicionar filtros adicionais conforme necessário.

Para obter mais informações sobre a escolha de um tipo de instância, consulte [Configurar uma frota de contêineres](#). O tamanho do tipo de instância que você escolher afetará a forma como os grupos de contêineres do servidor de jogos são agrupados em cada instância da frota. Dependendo da sua escolha, considere revisar sua configuração para os grupos de contêineres de servidores de jogos desejados por instância.

Etapa 4: revisar e criar

- Revise as configurações da sua frota.

Atualize os metadados da frota e a configuração a qualquer momento, independentemente do status da frota. Para obter mais informações, consulte [Atualizar uma configuração de Amazon GameLift Servers frota](#). É possível atualizar a capacidade da frota depois que ela atingir o status ACTIVE. Para obter mais informações, consulte [Escalando a capacidade de hospedagem de jogos com Amazon GameLift Servers](#). Também é possível adicionar ou remover locais remotos.

Quando terminar de revisar, escolha Criar.

Se sua solicitação for bem-sucedida, o console exibirá a página de detalhes do novo recurso da frota. Inicialmente, o status é NEW, como Amazon GameLift Servers inicia o processo de criação da frota. Você pode rastrear o status da nova frota na página Frotas. Uma frota está pronta para sediar sessões de jogo quando atinge o status ACTIVE.

AWS CLI

Para criar uma frota de contêineres com o AWS CLI, abra uma janela de linha de comando e use o `create-container-fleet` comando. Para obter mais informações sobre esse comando, consulte [create-container-fleet](#) na Referência de AWS CLI Comandos.

O exemplo de `create-container-fleet` solicitação mostrado abaixo cria uma nova frota de contêineres com as seguintes características:

- ContainerGroupsConfiguration Especifica apenas uma definição de grupo de contêineres de servidores de jogos: `MyAdventureGameContainerGroup`. O número de grupos de

contêineres de servidores de jogos que serão implantados em cada instância da frota é calculado por Amazon GameLift Servers.

- A frota usa instâncias sob demanda c5.large por padrão.
- Por padrão, a frota abre um conjunto de portas de conexão e portas de permissões de entrada, conforme calculado por Amazon GameLift Servers. Ele implanta grupos de contêineres nos seguintes locais:

```
aws gamelift create-container-fleet \  
  --fleet-role-arn arn:aws:iam::MyAccount:role/MyContainersRole \  
  --game-server-container-group-definition-name "rn:aws:gamelift:us-  
west-2:111122223333:containergroupdefinition/MyAdventureGameContainerGroup:2" \  

```

Se a solicitação de criação de frota for bem-sucedida, Amazon GameLift Servers retorna um conjunto de atributos da frota que inclui as configurações solicitadas e uma nova ID da frota de contêineres. Amazon GameLift Servers em seguida, define o status da frota e os status de localização como Novo e inicia o processo de ativação da frota. Você pode acompanhar o status da frota e visualizar outras informações da frota usando estes comandos da 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](#)

É possível alterar a capacidade da frota e outras definições de configuração, conforme necessário, usando estes comandos:

- [update-container-fleet](#)
- [update-fleet-capacity](#)
- [update-fleet-port-settings](#)
- [create-fleet-locations](#)
- [delete-fleet-locations](#)

Atualizar um Amazon GameLift Servers frota de contêineres gerenciada

Você pode atualizar a maioria das propriedades de uma frota de contêineres gerenciada, incluindo as definições do grupo de contêineres. Dependendo das configurações que estão sendo atualizadas, uma atualização da frota pode iniciar uma nova implantação da frota. Em uma implantação de frota, todas as instâncias da frota são removidas e substituídas por instâncias com a nova configuração. As configurações que exigem uma implantação incluem:

- Definições de grupos de contêineres, incluindo atualizações em imagens de contêineres
- Intervalos de portas de conexão e permissões de entrada
- Configuração de log

Você pode acompanhar o status das implantações da frota no [Amazon GameLift Servers console](#) ou o AWS Command Line Interface (AWS CLI) para criar uma frota de contêineres.

Console

No [Amazon GameLift Servers console](#), selecione Região da AWS onde você deseja criar a frota. As definições do grupo de contêineres devem estar na mesma região em que você deseja criar a frota.

Abra a barra de navegação esquerda do console e escolha Contêineres gerenciados: frotas. Na página de frotas de contêineres gerenciadas, selecione uma frota na lista e escolha Editar.

1. Atualize as configurações da frota de contêineres conforme necessário. Quando terminar, escolha Create (Criar).
2. Se suas atualizações exigirem a implantação de uma frota, você deverá especificar as opções de implantação da seguinte forma:
 - Proteção da sessão do jogo. Você pode optar por proteger as instâncias da frota que têm sessões de jogo ativas (implantação segura). Com essa configuração, as instâncias da frota não são substituídas até o final das sessões de jogo. Como alternativa, você pode optar por substituir as instâncias da frota, independentemente da atividade da sessão de jogo (implantação insegura). Implantações inseguras são úteis durante as fases de desenvolvimento e teste para reduzir o tempo de implantação.
 - Porcentagem mínima saudável. Você pode gerenciar a rapidez com que as instâncias da frota são substituídas. Use essa configuração para manter uma quantidade mínima

de tarefas íntegras durante a implantação. Um valor baixo prioriza a velocidade de implantação, enquanto um valor alto garante que a disponibilidade do servidor de jogos permaneça alta durante toda a implantação.

- **Estratégia de falha na implantação.** Decida quais ações tomar se uma implantação falhar. Uma falha na implantação significa que alguns dos contêineres atualizados falharam nas verificações de status e são considerados danificados. Você pode definir implantações para reverter automaticamente todas as instâncias da frota para o estado implantado anteriormente. Como alternativa, você pode optar por manter algumas das instâncias da frota danificadas para uso na depuração.

Se sua solicitação for bem-sucedida, o console exibirá a guia Implantações da frota de contêineres gerenciada. Use essa guia para rastrear o status de cada implantação. Se você iniciar uma nova implantação para a frota, essa ação cancelará automaticamente qualquer implantação que esteja atualmente em andamento para a frota.

AWS CLI

Para criar uma frota de contêineres com o AWS CLI, abra uma janela de linha de comando e use o `update-container-fleet` comando. Para obter mais informações sobre esse comando, consulte [update-container-fleet](#) na Referência de AWS CLI Comandos.

O exemplo a seguir atualiza uma frota de contêineres existente com as seguintes características:

- Ele atualiza a definição do grupo de contêineres do servidor de jogos para usar a versão 2.
- Ele especifica opções de implantação segura.

```
{
  "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"
}
```

Personalize uma frota de Amazon GameLift Servers contêineres

Os tópicos desta seção descrevem alguns dos recursos opcionais para contêineres Amazon GameLift Servers gerenciados. Você pode optar por usar qualquer um ou todos esses recursos.

Tópicos

- [Defina limites de recursos](#)
- [Designe recipientes essenciais](#)
- [Configurar conexões de rede](#)
- [Configurar verificações de saúde para contêineres](#)
- [Definir dependências de contêineres](#)
- [Configurar uma frota de contêineres](#)

Defina limites de recursos

Para cada grupo de contêineres, você pode determinar quanta memória e capacidade de computação o grupo de contêineres precisa para executar seu software. Amazon GameLift Servers depende dessas informações para gerenciar recursos em todo o grupo de contêineres. Ele também usa essas informações para calcular quantos grupos de contêineres de servidores de jogos uma imagem da frota pode conter. Você também pode definir limites para contêineres individuais.

Você pode definir um limite máximo de memória e capacidade de computação para um grupo de contêineres. Por padrão, esses recursos são compartilhados por todos os contêineres do grupo. Você pode personalizar ainda mais o gerenciamento de recursos definindo limites para contêineres individuais.

Defina limites opcionais para contêineres individuais

Definir limites de recursos específicos para contêineres permite que você exerça maior controle sobre como contêineres individuais podem usar os recursos do grupo. Se você não definir limites específicos para contêineres, todos os contêineres do grupo compartilharão os recursos do grupo. O compartilhamento oferece maior flexibilidade para usar os recursos onde eles são necessários. Também aumenta o potencial de os processos competirem entre si e resultarem em falhas no contêiner.

Defina qualquer uma das `ContainerDefinition` propriedades a seguir para qualquer contêiner.

- `MemoryHardLimitMebibytes`— Defina um limite máximo de memória para o contêiner. Se o contêiner exceder esse limite, isso resultará em uma reinicialização.
- `Vcpulimit` — Reserve uma quantidade mínima de recursos de vCPU para uso exclusivo do contêiner. O contêiner sempre tem o valor reservado disponível. Ele pode exceder esse mínimo a qualquer momento, se houver recursos adicionais disponíveis. (1024 unidades de CPU equivalem a 1 vCPU.)

Definir limites totais de recursos para um grupo de contêineres

Se você definir limites para contêineres individuais, talvez seja necessário modificar a quantidade de recursos de memória e vCPU que o grupo de contêineres precisa. O objetivo é alocar recursos suficientes para otimizar o desempenho do servidor de jogos. Amazon GameLift Servers usa esses limites para calcular como agrupar grupos de contêineres de servidores de jogos em uma instância de frota. Você também os usará ao escolher um tipo de instância para uma frota de contêineres.

Calcule a memória total e a vCPU necessárias para um grupo de contêineres. Considere o seguinte:

- Quais são todos os processos que são executados em todos os contêineres do grupo de contêineres? Some os recursos necessários para esses processos. Anote quaisquer limites específicos do contêiner.
- Quantos processos simultâneos de servidor de jogos você planeja executar em cada grupo de contêineres? Você determina isso na imagem do contêiner do seu servidor de jogos.

Com base em sua estimativa dos requisitos do grupo de contêineres, defina as seguintes `ContainerGroupDefinition` propriedades:

- `TotalMemoryLimitMebibytes`— Defina um limite máximo de memória para o grupo de contêineres. Todos os contêineres do grupo compartilham a memória alocada. Se você definir limites de contêineres individuais, o limite total de memória deverá ser igual ou maior que o limite máximo de memória específico do contêiner.
- `TotalVcpuLimit`— Defina um limite máximo de vCPU para o grupo de contêineres. Todos os contêineres do grupo compartilham os recursos de CPU alocados. Se você definir limites de contêineres individuais, o limite total de CPU deverá ser igual ou maior que a soma de todos os limites de CPU específicos do contêiner. Como prática recomendada, considere definir esse valor para dobrar a soma dos limites de CPU do contêiner.

Exemplo de cenário

Digamos que estamos definindo um grupo de contêineres de servidores de jogos com os três contêineres a seguir:

- O contêiner A é o nosso contêiner de servidor de jogos. Estimamos os requisitos de recursos para um servidor de jogos em 512 MiB e 1024 de CPU. Planejamos que o contêiner execute um processo de servidor. Como esse contêiner executa nosso software mais importante, não definimos limite de memória ou limite de reserva de vCPU.
- O contêiner B é executado em um contêiner de suporte com requisitos de recursos estimados em 1024 MiB e 1536 de CPU. Definimos um limite de memória de 2048 MiB e um limite de reserva de CPU de 1024 CPUs.
- O contêiner C é outro contêiner de suporte. Definimos um limite de memória rígida de 512 MiB e um limite de reserva de CPU de 512 CPUs.

Usando essas informações, definimos os seguintes limites totais para o grupo de contêineres:

- Limite total de memória: 7680 MiB. Esse valor excede o limite máximo de memória (1024 MiB).
- Limite total de CPU: 13312 CPU. Esse valor excede a soma do limite de CPU (1024+512 CPU).

Designe recipientes essenciais

Para um grupo de contêineres por instância, designe cada contêiner como essencial ou não essencial. Os grupos de contêineres por instância devem ter pelo menos um contêiner de suporte essencial. O contêiner essencial faz o trabalho crítico do grupo de contêineres. Espera-se que o contêiner essencial esteja sempre funcionando. Se falhar, todo o grupo de contêineres será reiniciado.

Defina `Essential` a `ContainerDefinition` propriedade como verdadeira ou falsa para cada contêiner.

Configurar conexões de rede

Você pode personalizar o acesso à rede para permitir que o tráfego externo se conecte a qualquer contêiner em uma frota de contêineres. Por exemplo, você deve estabelecer conexões de rede com o contêiner que executa os processos do servidor de jogos, para que os clientes do jogo possam entrar e jogar seu jogo. Clientes de jogos se conectam aos servidores de jogos usando portas e endereços IP.

Em uma frota de contêineres, a conexão entre cliente e servidor não é direta. Internamente, um processo em um contêiner escuta em uma porta de contêiner. Externamente, o tráfego de entrada se conecta a uma instância de frota usando uma porta de conexão. Amazon GameLift Servers mantém os mapeamentos entre as portas internas do contêiner e as portas de conexão externas, para que o tráfego de entrada seja roteado para o processo correto na instância.

Amazon GameLift Servers fornece uma camada extra de controle para suas conexões de rede. Cada frota de contêineres tem uma configuração de permissões de entrada, que permite controlar o acesso a cada porta de conexão externa. Por exemplo, você pode remover as permissões de todas as portas de conexão para desligar todo o acesso aos contêineres da frota.

Você pode atualizar as permissões de entrada, as portas de conexão e as portas de contêineres de uma frota.

Definir intervalos de portas de contêineres

Configure os intervalos de portas do contêiner como parte da definição de cada contêiner. Esse é um parâmetro obrigatório para a definição de um grupo de contêineres. Você precisa configurar portas suficientes para acomodar todos os processos em execução simultânea que precisam de acesso externo. Alguns contêineres não precisarão de portas.

Seu contêiner de servidor de jogos, que executa seus servidores de jogos, precisa de uma porta para cada processo de servidor de jogos em execução simultânea. O processo do servidor do jogo escuta a porta atribuída e a Amazon GameLift Servers reporta a.

Definir intervalos de portas de conexão

Configure sua frota de contêineres com um conjunto de portas de conexão. As portas de conexão fornecem acesso externo às instâncias da frota que estão executando seus contêineres. Amazon GameLift Servers atribui portas de conexão e as mapeia para portas de contêiner conforme necessário.

Por padrão, Amazon GameLift Servers calcula o número de portas necessárias para todos os grupos de contêineres e define um intervalo de portas para acomodá-los. É altamente recomendável usar valores Amazon GameLift Servers calculados, que são atualizados quando você implanta atualizações em uma definição de grupo de contêineres. Se você precisar personalizar os intervalos de portas de conexão, use as orientações a seguir.

Ao criar uma frota de contêineres, defina um intervalo de portas de conexão (consulte [ContainerFleet: InstanceConnectionPortRange](#)). Certifique-se de que o intervalo tenha portas

suficientes para mapear para cada porta de contêiner definida em todos os contêineres nos dois grupos de contêineres da frota. Para calcular as portas de conexão mínimas necessárias, use a seguinte fórmula:

```
[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]
```

Como prática recomendada, dobre o número mínimo de portas de conexão.

 Note

O número de portas de conexão pode potencialmente limitar o número de grupos de contêineres de servidores de jogos por instância. Se uma frota tiver apenas portas de conexão suficientes para um grupo de contêineres de servidores de jogos por instância, Amazon GameLift Servers implantará somente um grupo de contêineres de servidores de jogos, mesmo que as instâncias tenham capacidade computacional suficiente para vários grupos de contêineres de servidores de jogos.

Defina permissões de entrada

As permissões de entrada controlam o acesso externo a uma frota de contêineres especificando quais portas de conexão devem ser abertas para o tráfego de entrada. Você pode usar essa configuração para ativar e desativar o acesso à rede de uma frota conforme necessário.

Por padrão, Amazon GameLift Servers calcula o número de portas necessárias para todos os grupos de contêineres e define um intervalo de portas para acomodá-los. É altamente recomendável usar valores Amazon GameLift Servers calculados, que são atualizados quando você implanta atualizações em uma definição de grupo de contêineres. Se você precisar personalizar os intervalos de portas de conexão, use as orientações a seguir.

Ao criar uma frota de contêineres, defina um conjunto de permissões de entrada (consulte [ContainerFleet: InstanceInboundPermissions](#)). As portas de permissão de entrada devem corresponder aos intervalos de portas de conexão da frota.

Exemplo de cenário

Este exemplo ilustra como definir todas as três propriedades de conexão de rede.

- O grupo de contêineres de servidores de jogos da nossa frota tem 1 contêiner, que executa 1 processo de servidor de jogos.

Na definição do grupo de contêineres do servidor de jogos, definimos o `PortConfiguration` parâmetro para esse contêiner da seguinte forma:

```
"PortConfiguration": {  
  "ContainerPortRanges": [ { "FromPort": 10, "ToPort": 20, "Protocol": "TCP"} ]  
}
```

- Nossa frota também tem um grupo de contêineres por instância com 1 contêiner. Ele tem 1 processo que precisa de acesso à rede. Na definição de contêiner por instância, definimos o `PortConfiguration` parâmetro para esse contêiner da seguinte forma:

```
"PortConfiguration": {  
  "ContainerPortRanges": [ { "FromPort": 25, "ToPort": 25, "Protocol": "TCP"} ] }
```

- Nossa frota é configurada com 20 grupos de contêineres de servidores de jogos por instância da frota. Com essas informações, podemos usar a fórmula para calcular o número de portas de conexão necessárias:
 - Mínimo: 21 portas [1 porta de contêiner de servidor de jogos * 20 grupos de contêineres de servidores de jogos por instância + 1 porta de contêiner por instância]
 - Melhor prática: 42 portas [portas mínimas* 2]

Ao criar a frota de contêineres, definimos o `ConnectionPortRange` parâmetro da seguinte forma:

```
"ConnectionPortRange": { "FromPort": 1010, "ToPort": 1071 }
```

- Queremos permitir o acesso a todas as portas de conexão disponíveis. Ao criar a frota de contêineres, definimos o `InstanceInboundPermissions` parâmetro da seguinte forma:

```
"InstanceInboundPermissions": [  
  {"FromPort": 1010, "ToPort": 1071, "IpRange": "10.24.34.0/23", "Protocol":  
    "TCP"} ]
```

Configurar verificações de saúde para contêineres

Um contêiner é reiniciado automaticamente se ocorrer uma falha no terminal e parar de funcionar. Se um contêiner for considerado essencial, ele solicitará que todo o grupo de contêineres seja reiniciado.

Todos os contêineres do servidor de jogos são automaticamente considerados essenciais. Os contêineres de suporte podem ser designados como essenciais, mas precisam ter um mecanismo para relatar a integridade. Você também pode definir verificações de saúde para contêineres de suporte não essenciais.

Você pode definir critérios personalizados adicionais para medir a integridade do contêiner e usar uma verificação de saúde para testar esses critérios. Para configurar uma verificação de integridade do contêiner, você pode defini-la em uma imagem de contêiner do Docker ou em sua definição de contêiner. Se você definir uma verificação de saúde na definição do contêiner, ela substituirá todas as configurações na imagem do contêiner.

Defina as seguintes `SupportContainerDefinition` propriedades para uma verificação de integridade do contêiner:

- **Command**— Forneça um comando que verifique alguns aspectos da integridade do contêiner. Você decide quais critérios usar para medir a saúde. O comando deve resultar em um valor de saída de 1 (não íntegro) ou 0 (íntegro).
- **StartPeriod**— Especifique um atraso inicial antes que as falhas na verificação de integridade comecem a ser contadas. Esse atraso dá ao contêiner tempo para inicializar seus processos.
- **Interval**— Decida com que frequência executar o comando de verificação de integridade. Com que rapidez você deseja detectar e resolver uma falha no contêiner?
- **Timeout**— Decida quanto tempo esperar pelo sucesso ou pela falha antes de tentar novamente o comando de verificação de integridade. Quanto tempo o comando de verificação de integridade deve levar para ser concluído?
- **Retries**— Quantas vezes o comando de verificação de integridade deve ser repetido antes de registrar uma falha?

Definir dependências de contêineres

Em cada grupo de contêineres, você pode definir dependências entre contêineres com base no status do contêiner. Uma dependência afeta quando o contêiner dependente pode ser iniciado ou encerrado com base no status de outro contêiner.

Um caso de uso importante para dependências é criar sequências de inicialização e desligamento para o grupo de contêineres.

Por exemplo, você pode querer que o contêiner A comece primeiro e seja concluído com êxito antes do início dos contêineres B e C. Para conseguir isso, primeiro crie uma dependência para o contêiner B no contêiner A, com a condição de que o contêiner A seja concluído com êxito. Em seguida, crie uma dependência para o contêiner C no contêiner A com a mesma condição. As sequências de inicialização ocorrem na ordem inversa para o desligamento.

Configurar uma frota de contêineres

Ao criar uma frota de contêineres, considere os seguintes pontos de decisão. A maioria desses pontos depende da arquitetura e da configuração do contêiner.

Decida onde você deseja implantar sua frota

Em geral, você deseja implantar suas frotas geograficamente perto de seus jogadores para minimizar a latência. Você pode implantar sua frota de contêineres em qualquer Each Região da AWS que Amazon GameLift Servers suporte. Se você quiser implantar o mesmo servidor de jogo em outras localizações geográficas, você pode adicionar locais remotos à frota, incluindo Zonas Regiões da AWS Locais. Para uma frota com vários locais, você pode ajustar a capacidade de forma independente em cada localização da frota. Para obter mais informações sobre as localizações de frotas suportadas, consulte [Amazon GameLift Servers locais de serviço](#).

Escolha um tipo e tamanho de instância para sua frota

Amazon GameLift Servers suporta uma ampla variedade de tipos de EC2 instâncias da Amazon, todas disponíveis para uso com uma frota de contêineres. A disponibilidade e o preço do tipo de instância variam de acordo com o local. Você pode ver uma lista dos tipos de instância compatíveis, filtrados por localização, no Amazon GameLift Servers console (em Recursos, Instância e cotas de serviço).

Ao escolher um tipo de instância, primeiro considere a família de instâncias. As famílias de instâncias oferecem várias combinações de recursos de CPU, memória, armazenamento e rede. Obtenha mais informações sobre [famílias de EC2 instâncias](#). Em cada família, você tem uma variedade de tamanhos de instância para escolher. Considere os seguintes problemas ao selecionar um tamanho de instância:

- Qual é o tamanho mínimo de instância que pode suportar sua carga de trabalho? Use essas informações para eliminar qualquer tipo de instância que seja muito pequeno.

- Quais tamanhos de tipo de instância são adequados para sua arquitetura de contêiner? O ideal é escolher um tamanho que possa acomodar várias cópias do seu grupo de contêineres de servidores de jogos com o mínimo de espaço desperdiçado.
- Qual granularidade de escalabilidade faz sentido para seu jogo? A capacidade da frota de escala envolve adicionar ou remover instâncias, e cada instância representa a capacidade de hospedar um número específico de sessões de jogo. Considere a capacidade que você deseja adicionar ou remover com cada instância. Se a demanda dos jogadores variar milhares de minuto a minuto, talvez faça sentido usar instâncias muito grandes que possam hospedar centenas ou milhares de sessões de jogo. Por outro lado, você pode preferir um controle de escalabilidade mais refinado com tipos de instância menores.
- Há economia de custos disponível com base no tamanho? Você pode descobrir que o custo de certos tipos de instância varia de acordo com o local devido à disponibilidade.

Defina outras configurações opcionais da frota

Você pode usar os seguintes recursos opcionais ao configurar uma frota de contêineres:

- Configure seus servidores de jogos para acessar outros AWS recursos. Consulte [Comunique-se com outros AWS recursos de suas frotas](#).
- Proteja as sessões de jogo com jogadores ativos de serem encerradas prematuramente durante um evento de redução de escala.
- Limite o número de sessões de jogo que um indivíduo pode criar na frota em um período limitado de tempo.

Crie uma definição de grupo de contêineres para uma frota de Amazon GameLift Servers contêineres

Uma definição de grupo de contêineres descreve como implantar seus aplicativos de servidor de jogos em contêineres em uma frota de contêineres. É um plano que informa Amazon GameLift Servers quais imagens de contêiner devem ser implantadas na frota e como executá-las. Ao criar uma frota de contêineres, você especifica as definições do grupo de contêineres a serem implantadas na frota. Para obter mais informações sobre grupos de contêineres, consulte [componentes da frota de contêineres](#).

Antes de começar

Dicas sobre o que fazer antes de começar a criar uma definição de grupo de contêineres:

- Finalize suas imagens de contêiner e envie-as para um repositório do Amazon Elastic Container Registry (Amazon ECR) no mesmo Região da AWS local em que você planeja criar o grupo de contêineres. Amazon GameLift Servers captura um instantâneo de cada imagem no momento em que você cria a definição do grupo de contêineres e usa o instantâneo ao implantar em uma frota de contêineres. Consulte [Crie uma imagem de contêiner para Amazon GameLift Servers](#).
- Crie suas definições de contêiner como arquivos JSON. Uma definição de grupo de contêineres inclui uma ou mais definições de contêineres. Você pode usar os arquivos JSON se criar uma definição de grupo de contêineres usando o AWS CLI for Amazon GameLift Servers
- Verifique se seu AWS usuário tem permissões do IAM para acessar o repositório Amazon ECR. Consulte [Exemplos de permissão do IAM para Amazon GameLift Servers](#).

Crie uma definição de grupo de contêineres de servidores de jogos

Um grupo de contêineres de servidores de jogos executa seu software de servidor de jogos. Um grupo de contêineres de servidores de jogos tem um contêiner de servidor de jogos, que executa o executável do servidor de jogos. Ele também pode ter um ou mais contêineres de suporte para executar software adicional para dar suporte ao seu servidor de jogos. (Às vezes, são chamados de contêineres “auxiliares”).

Este tópico descreve como criar uma definição simples de grupo de contêineres de servidores de jogos usando o Amazon GameLift Servers console ou as ferramentas da AWS CLI. Para obter informações mais detalhadas sobre recursos opcionais, consulte [Personalize uma frota de Amazon GameLift Servers contêineres](#).

Note

Você pode alterar a maioria das definições de grupos e definições de contêineres depois de criá-las. Se você fizer alterações na definição de um contêiner, Amazon GameLift Servers captura um novo instantâneo das imagens atualizadas do contêiner.

Para criar uma definição simples de grupo de contêineres de servidores de jogos:

As instruções a seguir descrevem como criar uma definição de grupo de contêineres com os parâmetros mínimos necessários e usando os valores Amazon GameLift Servers padrão.

Console

No [Amazon GameLift Servers console](#), selecione Região da AWS onde você deseja criar o grupo de contêineres.

Abra a barra de navegação esquerda do console e escolha Contêineres gerenciados: definições de grupo. Na página de definição de grupos de contêineres, escolha Criar definição de grupo.

Etapa 1: Definir detalhes da definição do grupo de contêineres

1. Insira um nome de definição de grupo de contêineres. O nome deve ser exclusivo da região Conta da AWS e.
2. Selecione o tipo de grupo de contêineres do servidor de jogos.
3. Em Limite total de memória, insira o máximo de recursos de memória a serem disponibilizados para todos os contêineres no grupo de contêineres. Para obter ajuda no cálculo desse valor, consulte [Defina limites de recursos](#).
4. Em Limite total de vCPU, insira o poder computacional máximo a ser disponibilizado para todos os contêineres no grupo de contêineres. Para obter ajuda no cálculo desse valor, consulte [Defina limites de recursos](#).

Etapa 2: adicionar definições de contêiner

No mínimo, um grupo de contêineres de servidores de jogos tem um contêiner de servidor de jogos. No console, a primeira definição de contêiner que você cria é o contêiner do servidor de jogos. Esta etapa descreve como definir as configurações mínimas necessárias para uma definição de contêiner de servidor de jogos.

1. Insira um nome de definição de contêiner. Cada contêiner definido para o grupo deve ter um valor de nome exclusivo.
2. Vincule a uma imagem de contêiner com a versão do seu servidor de jogo. Insira o URI da imagem do Amazon ECR para uma imagem de contêiner em um repositório público ou privado. Você pode usar qualquer um dos seguintes formatos:
 - Somente URI da imagem: [Conta da AWS].dkr.ecr.[Região da AWS].amazonaws.com/[repository ID]
 - URI da imagem + resumo: [Conta da AWS].dkr.ecr.[Região da AWS].amazonaws.com/[repository ID]@[digest]

- URI da imagem + tag: [Conta da AWS].dkr.ecr.[Região da AWS].amazonaws.com/[repository ID]:[tag]
3. Especifique a versão Amazon GameLift Servers do SDK do servidor que a compilação do servidor do jogo usa. Para uma frota de contêineres, esse valor deve ser 5.2.0 ou superior.
 4. Em Intervalo de portas do contêiner interno, defina o protocolo e defina um intervalo de portas. O tamanho do intervalo deve ser maior que o número de processos simultâneos do servidor de jogos que serão executados nesse contêiner. Se o contêiner do servidor do jogo executar apenas um processo de servidor por contêiner, esse intervalo de portas precisará apenas de algumas portas. Consulte mais detalhes em [Configurar conexões de rede](#).
 5. Adicione mais contêineres conforme necessário para executar software de suporte adicional. Contêineres adicionais são automaticamente designados como contêineres de suporte. Um grupo de contêineres de servidores de jogos pode ter somente um contêiner de servidor de jogos e até oito contêineres de suporte. Forneça as seguintes configurações mínimas necessárias:
 - Nome da definição do contêiner
 - URI da imagem ECR.
 - Portas de contêiner internas (inclua isso somente se o contêiner tiver processos que precisem de acesso à rede).

Etapa 3: configurar dependências

- Se sua definição de grupo de contêineres tiver mais de um contêiner, você poderá, opcionalmente, definir dependências entre os contêineres. Para obter mais informações, consulte [Definir dependências de contêineres](#).

Etapa 3: revisar e criar

1. Revise todas as configurações de definição do seu grupo de contêineres. Use Editar para fazer alterações em qualquer seção, incluindo cada uma das definições de contêiner para o grupo.
2. Quando terminar de revisar, escolha Criar.

Se sua solicitação for bem-sucedida, o console exibirá a página de detalhes do novo recurso de definição de grupo de contêineres. Inicialmente, o status é `COPYING`, pois Amazon GameLift Servers começa a tirar instantâneos de todas as imagens de contêineres do grupo.

Quando essa fase for concluída, o status da definição do grupo de contêineres será alterado para READY. Uma definição de grupo de contêineres deve estar em READY status antes que você possa criar uma frota de contêineres com ela.

AWS CLI

Ao usar a AWS CLI para criar uma definição de grupo de contêineres, mantenha suas configurações de definição de contêiner em um arquivo separado. JSON Você pode referenciar o arquivo no comando da CLI. Consulte [Criar um JSON arquivo de definição de contêiner](#) para obter exemplos de esquemas.

Criar uma definição de grupo de contêineres

Para criar uma nova definição de grupo de contêineres, use o `create-container-group-definition` comando CLI. Para obter mais informações sobre esse comando, consulte [create-container-group-definition](#) na Referência de Comandos da AWS CLI.

Este exemplo ilustra uma solicitação para a definição de um grupo de contêineres de servidores de jogos. Ele pressupõe que você tenha criado um arquivo JSON com as definições de contêiner desse grupo.

```
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
```

Criar um **JSON** arquivo de definição de contêiner

Ao criar uma definição de grupo de contêineres, você também define os contêineres para o grupo. Uma definição de contêiner especifica o repositório Amazon ECR onde a imagem do contêiner é armazenada e configurações opcionais para portas de rede, limites de uso de CPU e memória e outras configurações. Recomendamos criar um único JSON arquivo com as configurações de todos os contêineres em um grupo de contêineres. A manutenção de um arquivo é útil para armazenar, compartilhar e rastrear versões dessas configurações críticas. Se você usar a AWS CLI para criar suas definições de grupo de contêineres, poderá referenciar o arquivo no comando.

Para criar uma definição de contêiner

1. Crie e abra um novo .JSON arquivo. Por exemplo:

```
[~/work/glc]$ vim SimpleServer.json
```

2. Crie uma definição de contêiner separada para cada um dos contêineres do grupo. Copie o conteúdo de exemplo a seguir e modifique-o conforme necessário para seus contêineres. Para obter detalhes sobre a sintaxe de uma definição de contêiner, consulte [ContainerDefinitionInput](#) Referência da Amazon GameLift Servers API.
3. Salve o arquivo localmente para que você possa consultá-lo em um comando da AWS CLI.

Exemplo: definição de contêiner de servidor de jogos

Example

Este exemplo descreve o contêiner essencial para seu grupo de contêineres de servidores de jogos. O contêiner de réplica essencial inclui seu aplicativo de servidor de jogos, o Amazon GameLift Servers Agente, e pode incluir outro software de suporte para sua hospedagem de jogos. A definição deve incluir um nome, URI de imagem e uma configuração de porta. Este exemplo também define alguns limites de recursos específicos do contêiner.

```
{
  "ContainerName": "MyAdventureGameServer",
  "ImageUri": "111122223333.dkr.ecr.us-east-1.amazonaws.com/gl-
containers:myadventuregame-server",
  "PortConfiguration": {
    "ContainerPortRanges": [
      {
        "FromPort": 2000,
        "Protocol": "TCP",
        "ToPort": 2010
      }
    ]
  },
  "ServerSdkVersion": "5.2.0"
}
```

Atualizar uma definição de grupo de contêineres para um Amazon GameLift Servers frota de contêineres

Você pode atualizar a maioria das propriedades de uma definição de grupo de contêineres, incluindo as definições de contêineres individuais. As definições de grupos de contêineres têm um número de versão. Quando você atualiza uma definição de grupo de contêineres, Amazon GameLift Servers salva a atualização e incrementa o número da versão da definição. Ao configurar uma frota de contêineres, você pode especificar qual versão de uma definição de grupo de contêineres deve ser implantada.

Depois de atualizar a definição de um grupo de contêineres, você pode implantar a nova versão em uma frota de contêineres nova ou existente.

Atualizar a definição de um grupo de contêineres do servidor de jogos

Este tópico descreve como atualizar a definição do grupo de contêineres do servidor de jogos usando o Amazon GameLift Servers console ou ferramentas AWS CLI. Para obter informações mais detalhadas sobre recursos opcionais, consulte [Personalize uma frota de Amazon GameLift Servers contêineres](#).

Para atualizar uma definição de grupo de contêineres:

Console

No [Amazon GameLift Servers console](#), selecione Região da AWS onde você deseja criar o grupo de contêineres.

Abra a barra de navegação esquerda do console e escolha Contêineres gerenciados: definições de grupos. Na página de definição de grupos de contêineres, escolha uma definição e uma versão do grupo de contêineres para atualizar.

Depois de salvar suas atualizações, você pode usar a nova versão para criar novas frotas de contêineres ou implantar as atualizações em uma frota de contêineres existente.

Etapa 1: Definir detalhes da definição do grupo de contêineres

- Você pode atualizar as configurações de limite total de memória e vCPU.

Etapa 2: adicionar definições de contêiner

Você pode fazer as seguintes atualizações de definição de contêiner:

- Atualize as definições de contêineres existentes.
 - Adicione novas definições de contêiner de suporte.
 - Remova as definições do contêiner de suporte.
1. Você pode atualizar o URI da imagem ECR. Certifique-se de atualizar a configuração da versão do SDK do servidor para corresponder à nova imagem.
 2. Você pode atualizar o intervalo de portas internas do contêiner conforme necessário. As alterações feitas nessas configurações podem afetar as configurações da porta de conexão de uma frota de contêineres quando essas alterações são implantadas em uma frota. Consulte mais detalhes em [Configurar conexões de rede](#).

Etapa 3: configurar dependências

- Você pode alterar as dependências conforme necessário. Para obter mais informações, consulte [Definir dependências de contêineres](#).

Etapa 3: revisar e criar

- Revise suas atualizações de definição de grupo de contêineres. Use Editar para fazer alterações adicionais em qualquer seção. Ao terminar, escolha Criar para gerar uma nova versão da definição do grupo de contêineres.

Se sua solicitação for bem-sucedida, o console exibirá a página de detalhes do novo recurso de definição de grupo de contêineres. Inicialmente, o status é `COPYING`, como Amazon GameLift Servers começa a tirar fotos de todas as imagens de contêineres do grupo. Quando essa fase for concluída, o status da definição do grupo de contêineres será alterado para `READY`. Uma definição de grupo de contêineres deve estar em `READY` status antes que você possa criar uma frota de contêineres com ela.

AWS CLI

Ao usar a AWS CLI para criar ou atualizar uma definição de grupo de contêineres, mantenha suas configurações de definição de contêiner em um arquivo separado. JSON Você pode referenciar o arquivo no comando da CLI. Consulte [Criar um JSON arquivo de definição de contêiner](#) para obter exemplos de esquemas.

Ao atualizar uma definição, você só precisa especificar os valores que deseja atualizar. Amazon GameLift Servers retém todos os valores que você não inclui na sua solicitação de atualização. Se você estiver alterando a definição de um contêiner. No entanto, ao alterar a definição de um contêiner, forneça um conjunto completo.

Para atualizar uma definição de grupo de contêineres

Para atualizar uma nova definição de grupo de contêineres, use o `update-container-group-definition` comando CLI. Para obter mais informações sobre esse comando, consulte [update-container-group-definition](#) na Referência de Comandos da AWS CLI.

Example : grupo de contêineres de servidores de jogos

Você pode especificar uma versão de definição de grupo de contêineres ao recuperar, atualizar ou excluir uma definição de grupo de contêineres ou ao criar ou atualizar uma frota de contêineres. Cada definição de grupo de contêineres tem uma propriedade de versão. Além disso, o valor ARN da definição e especifica o número da versão.

Este exemplo ilustra uma solicitação de alteração na definição de um grupo de contêineres do servidor de jogos. Ele pressupõe que você tenha criado um arquivo JSON com as definições de contêiner desse grupo. Este exemplo usa o valor ARN para o nome da definição e especifica que a atualização é para a versão 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
```

Clonar uma definição de grupo de contêineres

Você pode usar o Amazon GameLift Servers console para clonar uma definição de grupo de contêineres existente.

Para clonar um grupo de contêineres

1. No [Amazon GameLift Servers console](#), vá para o painel de navegação esquerdo e escolha Grupos de contêineres.
2. Na página de listagem de grupos de contêineres, selecione o grupo de contêineres existente que você deseja clonar. Depois de selecionar um grupo de contêineres, o botão Clonar fica ativo.
3. Escolha Clonar. Essa ação abre o assistente de criação de grupos de contêineres com configurações pré-preenchidas.
4. Insira um novo nome para o grupo de contêineres clonados. O grupo de contêineres na mesma região deve ter nomes exclusivos.
5. Percorra o grupo de contêineres e as páginas de definição de contêineres, revise e crie o novo grupo de contêineres.

Excluir uma definição de grupo de contêineres para um Amazon GameLift Servers frota de contêineres

Você tem várias opções para excluir uma definição de grupo de contêineres. Quando você exclui uma definição de grupo de contêineres, essa ação também exclui todas as definições de contêineres no grupo de contêineres.

As definições de grupos de contêineres podem ter várias versões. As versões do grupo de contêineres têm o mesmo nome, mas têm um número de versão diferente. A definição do grupo de contêineres ARNs especifica o nome e a versão.

Você pode especificar uma versão de definição de grupo de contêineres ao recuperar, atualizar ou excluir uma definição de grupo de contêineres ou ao criar ou atualizar uma frota de contêineres. Cada definição de grupo de contêineres tem uma propriedade de versão. Além disso, o valor ARN da definição especifica o número da versão.

Há várias maneiras de excluir definições de grupos de contêineres:

- Você pode excluir todas as versões de uma definição específica.
- Você pode excluir uma versão específica de uma definição específica.
- Você pode manter algumas das versões mais recentes e excluir as versões mais antigas de uma definição específica. Por exemplo, você pode excluir todas as versões anteriores à versão 5.

Você pode excluir uma versão de definição de grupo de contêineres somente se ela não estiver sendo usada em uma frota de contêineres.

Para excluir uma definição de grupo de contêineres

Console

No [Amazon GameLift Servers console](#), selecione Região da AWS onde você deseja criar o grupo de contêineres.

Abra a barra de navegação esquerda do console e escolha Contêineres gerenciados: definições de grupo. Na página Definições do grupo de contêineres, selecione a definição que você deseja modificar e escolha Excluir.

Você será solicitado a selecionar o tipo de exclusão que deseja fazer e especificar configurações adicionais, dependendo do tipo de exclusão.

AWS CLI

- Para excluir uma definição de grupo de contêineres, use o comando `delete-container-group-definition` CLI e forneça valores para o tipo de exclusão que você deseja fazer. Para obter mais informações sobre esse comando, consulte [delete-container-group-definition](#) na Referência de Comandos da AWS CLI.

Este exemplo ilustra uma solicitação para excluir todas as versões da definição de grupo de contêineres do servidor de jogos anteriores à versão 5.

Example

```
aws gamelift delete-container-group-definition \
  --name MyAdventureGameContainerGroup \
  --version-count-to-retain 5 \
```

Amazon GameLift Servers Frotas Anywhere

Use Anywhere frotas quando você quiser aproveitar Amazon GameLift Servers recursos com seus próprios recursos de hospedagem. As frotas Anywhere são comumente usadas como ambientes de teste para desenvolvimento iterativo ou junto com frotas gerenciadas em uma solução de hospedagem híbrida.

Uma frota Anywhere consiste em um conjunto de recursos computacionais (virtuais ou físicos) que você fornece e gerencia. Os recursos de computação podem estar em qualquer localização geográfica com conectividade, desde um laptop local até postos remotos. Ao configurar uma frota Anywhere, você adiciona computadores à frota registrando-os por meio de Amazon GameLift Servers. Cada computação é registrada com seu endereço IP (ou nome DNS) para que Amazon GameLift Servers pode estabelecer uma conexão com ele.

Implante o software de servidor de jogos em uma frota Anywhere instalando-o em cada recurso de computação e iniciando os processos do servidor de jogos. Cada processo de servidor de jogo lançado estabelece uma conexão com o Amazon GameLift Servers serviço e relata a prontidão para sediar uma sessão de jogo. Você pode usar as ferramentas existentes de gerenciamento de configuração e implantação para lidar com as tarefas iniciais de implantação e gerenciamento do host. Há algumas tarefas adicionais necessárias para uso com Amazon GameLift Servers, incluindo:

- Registrar e cancelar o registro de recursos de computação para adicioná-los ou removê-los da frota.
- Mantenha os tokens de up-to-date autenticação em todos os computadores. Os processos do servidor na computação os usam ao se conectar ao Amazon GameLift Servers serviço.

 Note

Opcionalmente, implante sua frota Anywhere com o Amazon GameLift Servers Agente para automatizar essas tarefas de gerenciamento de chaves. Consulte [Trabalhe com o Amazon GameLift Servers agente](#).

Consulte estes tópicos sobre como configurar e manter frotas Anywhere:

- [Roteiro de desenvolvimento para hospedagem com Anywhere Amazon GameLift Servers](#)
- [Roteiro de desenvolvimento para hospedagem híbrida com Amazon GameLift Servers](#)
- [Configure para desenvolvimento iterativo com Amazon GameLift Servers Anywhere](#)
- [Crie um Amazon GameLift Servers Frota em qualquer lugar](#)
- [Como funciona a criação de Amazon GameLift Servers frotas](#)
- [Atualizar uma configuração de Amazon GameLift Servers frota](#)

Fluxo de trabalho de criação da frota Anywhere

Para frotas em qualquer lugar, Amazon GameLift Servers configura somente o recurso da frota. Você configura e registra recursos de computação na frota, instala o software do servidor de jogos e inicia os processos do servidor de jogos para hospedar sessões de jogo.

1. Amazon GameLift Servers cria o recurso da frota na região de origem da frota. O status da frota e o status da localização personalizada estão definidos como Novo.
2. Amazon GameLift Servers começa a gravar eventos no registro de eventos da frota.
3. Depois que o recurso da frota for criado, Amazon GameLift Servers define o status da frota como Ativo. Nesse momento, você pode registrar novos recursos de computação na frota.

Crie um Amazon GameLift Servers Frota em qualquer lugar

Este tópico descreve como criar um Amazon GameLift Servers Frota em qualquer lugar. Com uma frota Anywhere, você pode usar o núcleo Amazon GameLift Servers recursos de gerenciamento de sessões de jogos ao hospedar sessões de jogo com seus próprios recursos computacionais. Crie uma frota Anywhere para o hardware on-premises ou outros recursos baseados em nuvem.

Em qualquer lugar, as frotas são comumente usadas ao lado Amazon GameLift Servers frotas gerenciadas em uma solução de hospedagem híbrida. Eles também fornecem ambientes de teste úteis ao desenvolver um jogo para hospedagem com Amazon GameLift Servers. Consulte esses tópicos para saber mais sobre quando e como incorporar Amazon GameLift Servers Qualquer lugar se transforma em uma solução de hospedagem de jogos:

- [Hospedagem híbrida](#)
- [Configurando uma frota de hospedagem com Amazon GameLift Servers](#)
- [Configure para desenvolvimento iterativo com Amazon GameLift Servers Anywhere](#)

Como as frotas Anywhere são autogerenciadas, a configuração de uma frota exige algum trabalho adicional. Para preparar uma frota Anywhere para hospedar sessões de jogo e jogadores, é necessário concluir as seguintes tarefas:

Tópicos

- [Antes de começar](#)
- [Criar uma política personalizada](#)

- [Criar uma frota Anywhere](#)
- [Adicionar um recurso de computação à frota](#)
- [Iniciar um servidor de jogos](#)

Antes de começar

Antes de criar uma frota Anywhere, execute as tarefas a seguir. Para obter orientações mais detalhadas, consulte [Roteiro de desenvolvimento para hospedagem com Anywhere Amazon GameLift Servers](#) ou [Roteiro de desenvolvimento para hospedagem híbrida com Amazon GameLift Servers](#).

- Integre o código do seu servidor de jogos com o Amazon GameLift Servers SDK do servidor versão 5.x (ou superior). Não é necessário concluir todas as tarefas de integração de jogos, apenas aquelas necessárias para a compilação de um servidor de jogos. Uma prática comum é configurar sua máquina local como uma frota Anywhere e usar uma interface de linha de comandos para testar a integração do servidor de jogos (consulte [Configure testes locais com o Amazon GameLift Servers Anywhere](#)). Você pode incorporar componentes adicionais (como Amazon GameLift Servers cliente de jogo ativado) à medida que você os desenvolve.
- Empacote o software de servidor de jogos para instalação nos recursos de computação da frota Anywhere. O pacote deve incluir a compilação integrada do servidor de jogos e todo o software de suporte necessário para executar o servidor.
- Decida se deseja usar o Amazon GameLift Servers Agente com sua frota Anywhere. O Agente é uma ferramenta de gerenciamento de processos no computador que automatiza algumas das principais tarefas relacionadas ao gerenciamento de processos e computações do servidor para uso com Amazon GameLift Servers. Para obter mais informações, consulte [Trabalhe com o Amazon GameLift Servers agente](#).

Criar uma política personalizada

Crie um local personalizado para representar o local físico dos recursos de computação. Ao criar uma frota Anywhere, é preciso ter pelo menos um local personalizado já definido. Você pode criar locais personalizados e adicioná-los a uma frota existente a qualquer momento.

Como criar um local personalizado

Use ou o Amazon GameLift Servers console ou o AWS Command Line Interface (AWS CLI) para criar um local personalizado.

Console

No [Amazon GameLift Servers console](#), use o painel de navegação para abrir a página Localizações. Escolha Criar local para abrir a caixa de diálogo Criar.

1. Na caixa de diálogo, insira o Nome do local. Como prática recomendada, use um nome que descreva um local significativo para um conjunto de recursos computacionais. Podem ser localizações geográficas, nome de um data center ou outro identificador de localização. Amazon GameLift Servers acrescenta o nome do seu local personalizado com custom-.
2. (Opcional) Adicione tags ao local personalizado. Cada tag consiste de uma chave e um valor opcional, que podem ser definidos. Atribua tags aos recursos da AWS que você deseja categorizar de maneiras úteis, como por finalidade, proprietário ou ambiente. Escolha Adicionar nova tag para cada tag a ser adicionada.
3. Escolha Criar.

AWS CLI

Crie um local personalizado usando o comando [create-location](#). Forneça um valor `location-name`, que deve começar com custom-. Como prática recomendada, use um nome que descreva um local significativo para um conjunto de recursos computacionais. Podem ser localizações geográficas, nome de um data center ou outro identificador de localização.

```
aws gamelift create-location \
  --location-name custom-location-1
```

Saída

```
{
  "Location": {
    "LocationName": "custom-location-1",
    "LocationArn": "arn:aws:gamelift:us-east-1:111122223333:location/custom-
location-1"
  }
}
```

Criar uma frota Anywhere

Crie uma frota Anywhere para um conjunto de recursos de computação que você possui. Uma nova frota Anywhere começa vazia; adicione recursos de computação à frota registrando-os.

Na criação, uma nova frota Anywhere passa rapidamente pelos status da frota de NEW para ACTIVE. Você pode adicionar recursos de computação à frota depois que ela atinge o status ACTIVE.

Como criar uma frota Anywhere

Use ou o Amazon GameLift Servers console ou o AWS Command Line Interface (AWS CLI) para criar uma frota Anywhere.

Console

No [Amazon GameLift Servers console](#), use o painel de navegação para abrir a página Fleets. Escolha Criar frota para iniciar o fluxo de trabalho de criação da frota.

Etapa 1 Escolha o tipo de computação

Selecione a opção Anywhere e escolha Avançar.

Etapa 2 Defina os detalhes da frota

Nessa etapa, especifique algumas configurações principais para toda a frota.

1. Preencha a seção Detalhes da frota:
 - a. Insira o Nome de uma frota. Recomendamos usar um padrão de nomenclatura de frota que facilite a identificação dos tipos de frota ao visualizar listas de frotas.
 - b. Forneça uma breve Descrição da frota.
2. Defina esses Detalhes adicionais opcionais conforme necessário. Você pode atualizar essas configurações da frota posteriormente.
 - a. Ao criar uma frota para testes de produção ou pré-produção, use essa configuração para especificar um valor de Custo por hora para os recursos de computação da frota. Amazon GameLift Servers pode usar essas informações durante o processo de colocação da sessão de jogo para selecionar recursos de hospedagem com base no custo.
 - b. Se você quiser combinar dados de métricas dessa frota e de outras, especifique um nome de Grupo de métricas. Use o mesmo nome de grupo de métricas para todas as

frotas que deseja combinar. Visualize as métricas do grupo de métricas para ver os dados agregados.

3. Adicione tags opcionais ao local personalizado. Cada tag consiste de uma chave e um valor opcional, que podem ser definidos. Atribua tags aos AWS recursos que você deseja categorizar de maneiras úteis, como por finalidade, proprietário ou ambiente. Escolha Adicionar nova tag para cada tag a ser adicionada.
4. Escolha Avançar para continuar o fluxo de trabalho.

Etapa 3 Selecione locais personalizados

Nessa etapa, identifique o local físico dos recursos de computação que você planeja adicionar a essa frota. É possível especificar um ou mais locais agora e adicionar ou remover locais posteriormente, conforme necessário.

1. Em Locais personalizados, selecione um ou mais locais para os recursos de computação da frota. A lista inclui todos os locais personalizados que foram definidos na Região da AWS selecionada no momento. Para definir um novo local personalizado que você deseja adicionar à frota, escolha Criar local.
2. Escolha Avançar para continuar o fluxo de trabalho.

Etapa 4 Revise e crie

Revise as configurações antes de criar a frota.

Quando tudo estiver pronto para implantar a nova frota, escolha Criar. Amazon GameLift Servers inicia imediatamente o processo de ativação da frota, atribuindo uma ID exclusiva e colocando a frota em NEW status. Você pode rastrear o progresso da frota na página Frotas.

AWS CLI

Use o comando [create-fleet](#) para criar uma frota com o tipo de recurso de computação ANYWHERE. Forneça um nome e pelo menos um local personalizado. Amazon GameLift Servers cria o recurso de frota Anywhere em seu padrão atual Região da AWS (ou você pode adicionar uma tag `--region` para especificar uma diferente Região da AWS).

O exemplo de solicitação a seguir cria uma frota com as configurações mínimas necessárias. Substitua *FleetName* e *custom-location* pelas próprias informações.


```
aws gamelift create-fleet \  
--name FleetName \  
--compute-type ANYWHERE \  
--locations "Location=custom-location"
```

Exemplo de resposta

```
{  
  "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"  
  }  
}
```

Na criação, uma nova frota Anywhere passa rapidamente para o status de frota `ACTIVE`. Você pode adicionar unidades de computação à frota depois que ela atinge o status `ACTIVE`.

Observe que a resposta não inclui os locais da frota. Você pode recuperar todos os detalhes da frota chamando [describe-fleet-attributes](#) e [describe-fleet-location-attributes](#).

Adicionar um recurso de computação à frota

Para adicionar um recurso de computação a uma frota e prepará-lo para hospedar sessões de jogo, execute as seguintes tarefas:

- Registre o recurso de computação com a frota. O registro diz Amazon GameLift Servers quais recursos de hospedagem física fazem parte da frota.

- Solicite um token de autenticação para o recurso de computação. Cada servidor de jogo executado no computador precisa desse token para se conectar ao Amazon GameLift Servers serviço. Os tokens de autenticação são temporários e devem ser atualizados regularmente.

Note

Se você estiver implantando seu software de servidor de jogos com o Amazon GameLift Servers Agente, você pode pular esta etapa. O Agent registra automaticamente cada recurso de computação e mantém um token de autenticação válido para o recurso de computação. Consulte [Trabalhe com o Amazon GameLift Servers agente](#).

Você pode registrar uma computação e solicitar um token de autenticação usando a AWS CLI ou fazendo chamadas programáticas para o SDK para AWS Amazon GameLift Servers. Essas ações não estão disponíveis por meio do Amazon GameLift Servers console.

Como prática recomendada, sugerimos automatizar essas duas tarefas adicionando um script de inicialização a cada recurso de computação. O script de inicialização chama automaticamente os comandos `register-compute` e `get-compute-auth-token`. Você também pode automatizar tarefas para atualizar regularmente o token de autenticação durante toda a vida útil do recurso de computação e cancelar o registro ao desligá-lo.

Cada uma das ações de inicialização retorna valores específicos do recurso de computação que é preciso armazenar nele. Quando um processo do servidor de jogos é iniciado no computador, ele deve passar esses valores como parâmetros do servidor ao inicializar uma conexão com o Amazon GameLift Servers serviço (veja [ServerParameters](#) na referência do SDK do servidor). Recomendamos que você defina esses valores específicos do recurso de computação (ou os locais armazenados) como variáveis de ambiente. Se você estiver usando o Amazon GameLift Servers Agente, essa tarefa é feita para você. Os valores específicos do recurso de computação são os seguintes:

- `register-compute` retorna um valor para `GameLiftServiceSdkEndpoint`. Defina esse valor para o parâmetro do servidor `WebSocketUrl`.
- `compute-auth-token` retorna o token de autenticação. Defina esse valor para o parâmetro do servidor `authToken`.

AWS CLI

As instruções a seguir descrevem como enviar manualmente cada solicitação usando a AWS CLI.

Como registrar um recurso de computação

Chame [register-compute](#) para registrar um recurso de computação. Identifique o ID da frota à qual adicionar o recurso de computação. Forneça as seguintes informações do recurso de computação: um nome significativo, endereço IP e localização. A localização do recurso de computação deve ser um local personalizado que já esteja associado à frota. Se você quiser usar um local personalizado diferente, use o Amazon GameLift Servers console para atualizar a frota ou chamar o comando AWS CLI [create-fleet-locations](#) para adicionar um local personalizado à frota.

No exemplo a seguir, substitua os valores de espaço reservado para o recurso de computação e a frota. O valor `fleet-id` é retornado quando você cria uma frota Anywhere. Você pode recuperar todos os detalhes da frota chamando [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
```

Exemplo de saída

```
{  
  "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"  
  }  
}
```

```
}
```

Como solicitar um token de autenticação

Chame [get-compute-auth-token](#) para solicitar um token de autenticação válido. Registre um recurso de computação. Identifique o ID da frota e o nome do recurso de computação.

No exemplo a seguir, substitua os valores de espaço reservado para a computação e a frota. O valor `fleet-id` é retornado ao criar-se uma frota Anywhere. É possível recuperar todos os detalhes da frota chamando [describe-fleet-attributes](#). Para encontrar informações dos recursos de computação, chame [list-compute](#) com o ID da frota para ver todos os recursos de computação registrados na frota.

```
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
```

Exemplo de saída

```
{  
  "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"  
}
```

Iniciar um servidor de jogos

Depois de criar uma frota Anywhere e adicionar um ou mais recursos de computação à frota, você estará com tudo pronto para começar a executar os servidores de jogos.

Etapas 1 Instale o software de servidor de jogos

Instale a compilação do servidor de jogos e todo o software dependente em cada recurso de computação da frota Anywhere. A construção do servidor do jogo deve ser integrada com

Amazon GameLift Servers SDK do servidor versão 5.x (ou superior) com a funcionalidade mínima necessária para se comunicar com o Amazon GameLift Servers serviço.

Etapa 2 Prepare os recursos de computação para executar um servidor de jogos

Certifique-se de que cada recurso de computação esteja registrado e tenha um token de autenticação válido. Se estiver usando scripts para gerenciar essas tarefas, garanta que os scripts sejam executados em cada recurso de computação antes de iniciar qualquer processo do servidor de jogos.

Se você implantou o Amazon GameLift Servers Agente com seu software de servidor de jogos, certifique-se de que o executável do Agente seja iniciado.

Etapa 3 Inicie um processo de servidor de jogos

Execute uma instância do executável do servidor de jogos em uma computação. Se a compilação do servidor de jogos estiver devidamente integrada, o processo do servidor de jogos chamará a ação `InitSDK()` do SDK do servidor com um conjunto de parâmetros de servidor válidos. Quando o processo de servidor estiver pronto para hospedar uma sessão de jogo, ele chamará `ProcessReady()`.

Note

Se você implantou seu software de servidor de jogos com o Amazon GameLift Servers Agente, você pode pular esta etapa. O Agente inicia automaticamente os processos do servidor de jogos com base nas instruções de runtime fornecidas por você.

É possível monitorar o progresso visualizando as métricas do processo do servidor para processos de servidor em ativação e ativos. Consulte [Métricas do Amazon GameLift Servers para frotas](#). Se o processo do servidor de jogos falhar ao inicializar, verifique se o processo está recuperando os valores corretos dos parâmetros do servidor para o recurso de computação em que está sendo executado.

Abstrato e Amazon GameLift Servers designação de frota com um alias

Uma Amazon GameLift Servers O alias é usado para abstrair um destino de hospedagem. Os destinos de hospedagem contam Amazon GameLift Servers onde procurar recursos disponíveis para hospedar uma nova sessão de jogo para jogadores. Os aliases são úteis nos seguintes cenários:

- Se seu jogo não usa filas de várias frotas para posicionamento de sessões de jogo, ele solicita novas sessões de jogo especificando um Amazon GameLift Servers ID da frota. Durante o ciclo de vida de um jogo, você substituirá a frota várias vezes para atualizar a compilação de um servidor, atualizar o hardware e o sistema operacional de hospedagem ou resolver problemas de desempenho. Use um alias para resumir o ID da frota, permitindo que você alterne facilmente o tráfego de jogadores de uma frota existente para a nova.
- Se você deseja fazer algo além de criar uma sessão de jogo quando um cliente de jogo faz uma solicitação. Por exemplo, talvez você queira direcionar jogadores que estão usando um out-of-date cliente para um site de upgrade.

Um alias deve especificar uma estratégia de roteamento. Existem dois tipos. Uma estratégia de roteamento simples direciona o tráfego do jogador para um ID de frota especificado, que você pode atualizar para redirecionar o tráfego. Uma estratégia de roteamento de terminal envia uma mensagem de volta para o cliente em vez de criar uma sessão de jogo. É possível alterar a estratégia de roteamento de um alias a qualquer momento.

Se você usa uma fila para posicionamento de uma sessão de jogo, não é necessário um alias para redirecionar o tráfego ao substituir uma frota. Com uma fila, você pode simplesmente adicionar a nova frota e remover a frota antiga. Essa ação não é visível para os jogadores, porque as novas solicitações de sessão de jogo são atendidas automaticamente usando a nova frota. Isso não afeta as sessões de jogo existentes. Você pode identificar destinos de filas usando um ID de frota ou um alias.

Tópicos

- [Crie um Amazon GameLift Servers alias](#)
- [Editar um alias](#)

Crie um Amazon GameLift Servers alias

Este tópico descreve como criar um Amazon GameLift Servers alias para uso com o posicionamento da sessão do jogo.

Para criar um alias

Use ou o Amazon GameLift Servers console ou o AWS Command Line Interface (AWS CLI) para criar um alias.

Console

No [Amazon GameLift Servers console](#), use o painel de navegação para abrir a página Aliases.

1. Escolha Criar alias.
2. Insira um Nome de alias. Recomendamos incluir características significativas no nome do alias para facilitar a visualização de uma lista de aliases.
3. Insira uma Descrição do alias conforme necessário.
4. Escolha uma Estratégia de roteamento para o alias.
 - a. Se você escolher uma estratégia de roteamento simples, selecione um ID de frota na lista para associar a esse alias. A lista inclui todas as frotas atualmente selecionadas. Região da AWS É necessário criar um alias na mesma região da frota.
 - b. Se você escolher uma estratégia de roteamento do Terminal, insira um valor de string que você deseja Amazon GameLift Servers para retornar a um cliente de jogo em resposta a uma solicitação de sessão de jogo. Uma solicitação com um alias de terminal gera uma exceção com a mensagem incorporada.
5. (Opcional) Adicione tags ao recurso de alias. Cada tag consiste de uma chave e um valor opcional, que podem ser definidos. Atribua tags aos AWS recursos que você deseja categorizar de maneiras úteis, como por finalidade, proprietário ou ambiente. Escolha Adicionar nova tag para cada tag a ser adicionada.
6. Quando estiver pronto para implantar a nova frota, escolha Criar.

AWS CLI

Use o [create-alias](#) comando para criar um alias. Amazon GameLift Servers cria o recurso de alias em seu padrão atual Região da AWS (ou você pode adicionar uma tag `--region` para especificar uma diferente Região da AWS).

No mínimo, inclua um nome de alias e uma estratégia de roteamento. Para uma estratégia de roteamento simples, especifique o ID de uma frota na mesma região do alias. Para uma estratégia de roteamento de terminal, forneça uma string de mensagens.

Editar um alias

Você pode editar um alias usando o Amazon GameLift Servers [console](#) ou com o comando [AWS CLI update-alias](#).

Este tópico descreve como editar um Amazon GameLift Servers alias para uso com o posicionamento da sessão do jogo. Você pode fazer as seguintes edições:

Como editar um alias

Use ou o Amazon GameLift Servers console ou o AWS Command Line Interface (AWS CLI) para editar um alias.

Console

No [Amazon GameLift Servers console](#), use o painel de navegação para abrir a página Aliases.

1. Selecione o alias que deseja editar e selecione Editar. Se você não vê o alias que deseja editar, verifique o que está atualmente selecionado Região da AWS.
2. Na página Editar alias, você pode fazer as seguintes edições:
 - Altere o nome do alias.
 - Altere a descrição do alias.
 - Mude a estratégia de roteamento de simples para terminal ou de terminal para simples.
 - Para um alias com uma estratégia de roteamento simples, altere o ID da frota ao qual o alias está associado.
 - Para um alias com uma estratégia de roteamento de terminal, altere o texto da mensagem.
3. Escolha Salvar alterações. Ao atualizar o ID da frota para um alias com uma estratégia de roteamento simples, a transição pode levar até 2 minutos para ser concluída. Durante esse período, novos posicionamentos de sessões de jogo podem ocorrer na frota antiga.

AWS CLI

Use o comando [update-alias](#) para fazer alterações em um recurso de alias. Você pode atualizar um recurso de alias em seu padrão Região da AWS atual ou adicionar uma `--region` tag para especificar uma diferente Região da AWS.

É possível alterar as seguintes propriedades:

- Nome do alias.
- Descrição do alias.
- Tipo de estratégia de roteamento. Forneça um ID da frota ou uma string de mensagens para a nova estratégia de roteamento.
- ID de frota para uma estratégia de roteamento simples existente. O ID da frota deve estar na mesma região que o alias.
- A string de mensagens para uma estratégia de roteamento de terminal existente.

Gerenciando o posicionamento da sessão de jogo com Amazon GameLift Servers filas

Uma fila de sessões de jogo é o principal mecanismo que Amazon GameLift Servers usa para pesquisar servidores de jogos disponíveis e escolhê-los para hospedar novas sessões de jogo. As filas oferecem uma maneira muito mais eficiente de processar um grande número de solicitações de sessões de jogos e encontrar canais para elas em várias frotas de recursos de hospedagem. Se sua solução de hospedagem usa mais de uma frota e você está processando grandes volumes de solicitações, provavelmente precisará de uma fila.

Quando seu jogo quer iniciar uma nova sessão de jogo para jogadores, ele envia uma solicitação de posicionamento para o Amazon GameLift Servers serviço, que o direciona para a fila. A configuração da fila determina quando e como as solicitações são processadas. Ao processar uma solicitação de colocação, Amazon GameLift Servers pesquisa um conjunto de frotas em busca de um servidor de jogos para hospedar a sessão de jogo. A colocação é bem-sucedida quando Amazon GameLift Servers encontra um servidor de jogo disponível e solicita que ele inicie uma sessão de jogo.

Tópicos

- [Características da fila](#)
- [Criar uma fila de sessões de jogo](#)
- [Personalize uma fila de sessões de jogo](#)
- [Configurar notificações de eventos para o posicionamento de sessões de jogo](#)
- [Tutorial: Crie uma Amazon GameLift Servers fila com instâncias spot](#)

Características da fila

Uma Amazon GameLift Servers a fila de sessões de jogos é um recurso AWS na nuvem. Você pode criar uma fila em qualquer Região da AWS uma Amazon GameLift Servers suportes (consulte [Amazon GameLift Servers locais de serviço](#)). As solicitações de colocação de sessões de jogo são enviadas para esse local e processadas lá.

Automatizar o posicionamento das sessões de jogo com filas oferece benefícios significativos tanto para desenvolvedores quanto para jogadores. Isso inclui:

- As filas oferecem o “melhor posicionamento possível”. Ao processar solicitações de posicionamento de sessão de jogo, uma fila usa o Amazon GameLift Servers Algoritmo FleetiQ para priorizar posicionamentos com base em um conjunto de preferências definidas, incluindo custo, localização e latência do player.
- As filas oferecem suporte às frotas Spot para ajudar a reduzir os custos de hospedagem de jogos. Você pode configurar suas filas com frotas AWS spot, que geralmente oferecem custos de hospedagem significativamente mais baixos, bem como frotas sob demanda. Como o baixo custo é um dos principais critérios para colocações, as filas sempre podem aproveitar as diferenças de custo.
- As filas podem colocar novos jogos mais rapidamente durante a alta demanda. Ao configurar uma fila com várias frotas, você oferece opções mais flexíveis para a colocação de sessões de jogo. Mas frotas adicionais também fornecem capacidade de backup conforme necessário quando a demanda aumenta. Para qualquer solicitação de colocação, se Amazon GameLift Servers não consegue colocar uma sessão de jogo no local preferido, ela avança automaticamente para avaliar outros locais.
- As filas podem tornar a disponibilidade do servidor de jogos mais resiliente. Interrupções podem acontecer. Com uma fila de várias frotas, uma desaceleração ou interrupção não precisa afetar o acesso do jogador ao seu jogo. Ao configurar sua fila com frotas com capacidade em zonas diferentes Regiões da AWS e de disponibilidade, você pode ajudar a garantir que os jogadores sempre encontrem uma sessão de jogo para participar.
- Obtenha métricas sobre posicionamentos de sessões de jogo e desempenho de fila. Amazon GameLift Servers emite métricas de fila, incluindo estatísticas sobre sucessos e falhas de posicionamento, o número de solicitações na fila e o tempo médio que as solicitações passam na fila. Você pode ver essas métricas no Amazon GameLift Servers console ou entrada CloudWatch.

Para começar criando uma fila inicial básica, consulte. [Criar uma fila de sessões de jogo](#)

Criar uma fila de sessões de jogo

As filas são usadas para colocar novas sessões de jogo em várias frotas e locais. Seu jogo inicia novas sessões de jogo enviando solicitações de colocação para uma fila. Uma fila é configurada com instruções sobre como processar solicitações. Saiba mais sobre como iniciar solicitações de colocação de sessões de jogo em [Criar sessões de jogos](#).

Para criar uma fila de sessões de jogo

Essas instruções ilustram como criar uma fila de trabalho simples com configurações mínimas e configurações padrão. Há várias opções para personalizar a configuração de uma fila. Essas opções ajudam você a fazer as melhores colocações possíveis com base nas necessidades do seu jogo. Para saber mais sobre como personalizar filas para seu jogo, consulte [Personalize uma fila de sessões de jogo](#). Você pode atualizar a maioria das configurações de fila a qualquer momento.

Você pode criar uma fila de sessões de jogo usando o Amazon GameLift Servers console ou a AWS CLI.

Console

No [Amazon GameLift Servers console](#), selecione uma AWS região para trabalhar. Abra a barra de navegação esquerda do console e escolha Filas.

1. Na página Filas, escolha Criar fila para iniciar o fluxo de trabalho.
2. Em Configurações da fila, insira as seguintes configurações:
 - a. Insira um nome de fila. Esse nome deve ser exclusivo daquele em Região da AWS que você está criando a fila.
 - b. Mantenha a configuração de tempo limite padrão, que é de 600 segundos (ou 10 minutos). Esse valor controla por quanto tempo Amazon GameLift Servers tenta colocar uma nova sessão de jogo antes de parar. Amazon GameLift Servers pesquisa os recursos disponíveis até que a solicitação atinja o tempo limite. Você pode atualizar a configuração de tempo limite de uma fila a qualquer momento.
 - c. Ignore a seção Políticas de latência do player. Uma fila usa políticas de latência somente quando recebe solicitações de posicionamento que incluem dados de latência do jogador. Você pode adicionar políticas de latência a uma fila a qualquer momento. Para saber mais sobre a criação de políticas de latência, consulte [Crie uma política de latência do jogador](#).
3. Pule a seção Locais de posicionamento da sessão de jogo para usar a configuração padrão de Todos os locais. Essa configuração permite criar uma lista de permissões de locais em que a fila pode fazer posicionamentos (também chamada de configuração de filtro). Para obter mais informações sobre a priorização por localização e configurações de filtro, consulte [Priorize os posicionamentos por localização](#).
4. Em Ordem de destino, adicione uma ou mais frotas à fila. Você pode identificar frotas usando fleet IDs ou ARNs, ou usando um alias de frota. Ao adicionar várias frotas, lembre-se de que todas elas devem estar executando versões de jogo semelhantes e ser compatíveis com

qualquer cliente de jogo que use essa fila. Além disso, todas as frotas em uma fila devem ter a mesma configuração de certificado.

- a. Selecione a região em que a frota ou o alias foi criado. Para uma frota com vários locais, essa é a região “inicial”.
 - b. Em Tipo de destino, selecione uma frota ou um alias.
 - c. Suas seleções de região e tipo preenchem uma lista suspensa de frotas ou aliases existentes. Selecione um para designar como destino da fila.
 - d. Para especificar outra frota ou alias para a fila, escolha Adicionar destino e repita as etapas anteriores.
 - e. Depois de adicionar uma lista de destinos, use o drag-and-drop recurso para reordenar os destinos. Amazon GameLift Servers usa essa ordem ao priorizar os posicionamentos por destino.
5. Ignore a seção Prioridade de posicionamento da sessão de jogo para manter a ordem de prioridade padrão. Essa configuração permite que você personalize como Amazon GameLift Servers escolhe onde procurar os recursos de hospedagem disponíveis para novas colocações em sessões de jogo. Para obter mais informações sobre como priorizar posicionamentos, consulte [Priorize o posicionamento de sessões de jogo](#). Você pode atualizar as prioridades de posicionamento de uma fila a qualquer momento.
 6. Em Ordem de localização, mantenha os valores padrão. Essa configuração é usada ao priorizar por localização da frota. Ele fornece a ordem de localização para uso. Ao usar as configurações de prioridade padrão, a localização é usada como critério de desempate quando o destino preferido é uma frota com vários locais.
 7. Ignore a seção opcional de configurações de notificação de eventos. As notificações de eventos são necessárias para filas que processam um grande volume de solicitações de posicionamento. Para filas que processam volumes baixos, como para fins de desenvolvimento ou teste, você pode acompanhar o status das solicitações de posicionamento fazendo uma pesquisa com [DescribeGameSessionPlacement](#). Consulte mais detalhes em [Configurar notificações de eventos para o posicionamento de sessões de jogo](#). Você pode atualizar as configurações de notificação de eventos de uma fila a qualquer momento.
 8. Escolha Criar para gerar uma nova fila com o mínimo de personalização.

AWS CLI

Example Criar uma fila

O exemplo a seguir cria uma fila de sessão de jogo com estas configurações:

- Um tempo limite de cinco minutos.
- Dois destinos de frota.
- Filtre para permitir apenas colocações nesses locais:us-east-1,us-east-2. us-west-2, ca-central-1 e.
- Pedido prioritário com base no custo e, em seguida, nos locais em um pedido especificado.

```
aws gamelift create-game-session-queue \  
  --name "sample-test-queue" \  
  --timeout-in-seconds 300 \  
  --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

Você pode obter valores de ARN da frota e do alias chamando [ou descrevendo o alias com o ID da frota describe-fleet-attributes](#) ou do alias.

Se a create-game-session-queue solicitação for bem-sucedida, Amazon GameLift Servers retornará um [GameSessionQueue](#) objeto com a nova configuração de fila. Agora você pode enviar solicitações para a fila usando [StartGameSessionPlacement](#).

Example Criar uma fila com políticas de latência de jogador

O exemplo a seguir cria uma fila de sessões de jogo com estas configurações:

- Um tempo limite de dez minutos

- Três destinos de frota
- Um conjunto de políticas de latência dos jogadores

```
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 a `create-game-session-queue` solicitação for bem-sucedida, Amazon GameLift Servers retornará um [GameSessionQueue](#) objeto com a nova configuração de fila.

Personalize uma fila de sessões de jogo

Este tópico descreve como personalizar suas filas de sessões de jogo para tomar as melhores decisões possíveis sobre o posicionamento das sessões de jogo. Para obter mais informações sobre filas de sessões de jogo e como elas funcionam, consulte [Gerenciando o posicionamento da sessão de jogo com Amazon GameLift Servers filas](#).

Esses Amazon GameLift Servers recursos exigem filas:

- [Como marcar jogos com o FlexMatch](#)
- [Criar uma fila para instâncias spot](#)

Tópicos

- [Práticas recomendadas para filas de sessões de Amazon GameLift Servers jogos](#)
- [Definir o escopo de uma fila](#)
- [Criar uma fila de vários locais](#)

- [Priorize o posicionamento de sessões de jogo](#)
- [Avaliar métricas de fila](#)
- [Criar uma fila para instâncias spot](#)

Práticas recomendadas para filas de sessões de Amazon GameLift Servers jogos

Uma fila de sessões de jogo contém uma lista de frotas onde Amazon GameLift Servers podem colocar novas sessões de jogo. Cada frota pode ter recursos de hospedagem implantados em várias localizações geográficas. Ao escolher um posicionamento, a fila seleciona uma frota e uma localização da frota com base em um conjunto de prioridades que você definiu para a frota.

Considere as seguintes diretrizes e melhores práticas:

- Adicione frotas em locais que cubram seus jogadores. Você pode adicionar frotas e aliases em qualquer local disponível. A localização é importante se você estiver fazendo posicionamentos com base na latência relatada do jogador.
- Use aliases para todas as frotas. Atribua um alias a cada frota em uma fila e use os nomes de alias ao definir destinos em sua fila.
- Use a mesma versão ou script de jogo ou similar para todas as frotas. A fila pode colocar os jogadores em sessões de jogo em qualquer frota na fila. Os jogadores devem poder jogar em qualquer sessão de jogo em qualquer frota.
- Crie frotas em pelo menos dois locais. Ao ter servidores de jogos hospedados em pelo menos um outro local, você atenua o impacto das interrupções regionais em seus jogadores. Você poderá reduzir a escala de suas frotas de backup e usar o ajuste de escala automático para aumentar a capacidade se o uso aumentar.
- Priorize o posicionamento da sessão de jogo. Uma fila prioriza as opções de posicionamento com base em vários elementos, incluindo a ordem da lista de destinos.
- Crie sua fila no mesmo local do seu atendimento ao cliente. Ao colocar sua fila em um local próximo ao atendimento ao cliente, você pode minimizar a latência de comunicação.
- Use frotas com vários locais. Use a configuração do filtro de fila para evitar que a fila coloque sessões de jogo em locais específicos. Você poderá usar pelo menos duas frotas de vários locais com residências diferentes para mitigar o impacto das colocações de jogos durante uma interrupção regional.

- Use a mesma configuração de certificado TLS para todas as frotas. Os clientes de jogos que se conectam às sessões de jogo em suas frotas devem ter protocolos de comunicação compatíveis.

Definir o escopo de uma fila

A população de jogadores do seu jogo pode ter grupos de jogadores que não deveriam jogar juntos. Por exemplo, se você publicar seu jogo em dois idiomas, cada idioma deverá ter seus próprios servidores de jogos.

Para configurar o posicionamento de sessões de jogo para sua população de jogadores, crie uma fila separada para cada segmento de jogadores. Avalie cada fila para posicionar os jogadores nos servidores de jogos corretos. Algumas formas comuns de definir o escopo das filas incluem:

- Por localizações geográficas. Ao implantar seus servidores de jogos em várias áreas geográficas, você pode criar filas para jogadores em cada local a fim de reduzir a latência dos jogadores.
- Por variações de compilação ou script. Se tiver mais de uma variação do seu servidor de jogos, talvez você esteja apoiando grupos de jogadores que não podem jogar nas mesmas sessões de jogo. Por exemplo, compilações ou scripts de servidores de jogos podem oferecer suporte a diferentes idiomas ou tipos de dispositivos.
- Por tipos de eventos. Você pode criar uma fila especial para gerenciar jogos para participantes de torneios ou outros eventos especiais.

Criar várias filas

Dependendo do jogo e dos jogadores, talvez você queira criar mais de uma fila de sessões de jogo. Quando o serviço do cliente do jogo solicita uma nova sessão de jogo, ele especifica que fila da sessão de jogo usar. Para ajudá-lo a determinar se você deve usar várias filas, considere:

- Variações do servidor de jogos. Você pode criar uma fila separada para cada variação do servidor de jogos. Todas as frotas de uma fila devem implantar servidores de jogos compatíveis. Isso ocorre porque os jogadores que usam a fila para entrar nos jogos devem poder jogar em qualquer um dos servidores de jogos da fila.
- Grupos de jogadores diferentes. Você pode personalizar a forma como Amazon GameLift Servers organiza as sessões de jogo com base no grupo de jogadores. Por exemplo, você pode precisar de filas personalizadas para determinados modos de jogo que exigem um tipo especial de instância ou de configuração de runtime. Ou, talvez, você queira uma fila especial para gerenciar posicionamentos em um torneio ou outro evento.

- Métricas da fila de sessões de jogo. Você pode configurar filas com base em como deseja coletar as métricas de posicionamento de sessões de jogo. Para obter mais informações, consulte [Métricas do Amazon GameLift Servers para filas](#).

Criar uma fila de vários locais

Recomendamos um design de vários locais para todas as filas. Esse design pode melhorar a velocidade do posicionamento e a resiliência da hospedagem. É necessário um design de vários locais para usar os dados de latência do jogador e posicionar os jogadores em sessões de jogo com latência mínima. Se estiver criando filas de vários locais que usam frotas de instâncias spot, siga as instruções em [Tutorial: Crie uma Amazon GameLift Servers fila com instâncias spot](#).

Uma forma de criar uma fila de vários locais é adicionar uma [frota de vários locais](#) a uma fila. Dessa forma, a fila pode posicionar sessões de jogo em qualquer um dos locais da frota. Você também pode adicionar outras frotas com configurações ou locais residenciais diferentes para redundância. Se você estiver usando uma frota de instâncias spot de vários locais, siga as melhores práticas e inclua uma frota de instância sob demanda com os mesmos locais.

O exemplo a seguir descreve o processo de criação de uma fila básica de vários locais. Neste exemplo, usamos duas frotas: uma frota de instâncias spot e uma frota de instâncias sob demanda. Cada frota tem o seguinte Regiões da AWS para locais de colocação: `us-east-1` `us-east-2` `central-1`, `us-west-2` e.

Para criar uma fila básica de vários locais com frotas de vários locais

1. Escolha um local para criar a fila. Você pode minimizar a latência da solicitação colocando a fila em um local próximo ao local onde você implantou o serviço ao cliente. Neste exemplo, criamos a fila em `us-east-1`.
2. Crie uma nova fila e adicione suas frotas de vários locais como destinos de fila. A ordem de destino determina como Amazon GameLift Servers organiza as sessões de jogo. Neste exemplo, listamos a frota de instâncias spot primeiro e a frota de instâncias sob demanda em segundo lugar.
3. Defina a ordem de prioridade de posicionamento de sessões de jogo da fila. Essa ordem determina onde a fila procura primeiro por um servidor de jogos disponível. Neste exemplo, usamos a ordem de prioridade padrão.
4. Defina a ordem dos locais. Se você não definir a ordem dos locais, Amazon GameLift Servers use os locais em ordem alfabética.

Game session placement locations

Locations where the queue can place new game sessions.

Locations

Choose locations

ca-central-1

X

Canada (Central)

us-west-2

X

US West (Oregon)

us-east-2

X

US East (Ohio)

us-east-1

X

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

Priorize o posicionamento de sessões de jogo

Amazon GameLift Servers usa um algoritmo para determinar como priorizar os destinos de uma fila e determinar onde colocar uma nova sessão de jogo. O algoritmo é baseado em um conjunto ordenado de critérios. Você pode usar a ordem de prioridades padrão ou personalizá-la. Você pode editar a ordem prioritária de uma fila a qualquer momento.

Ordem de prioridades padrão

1. **Latência** — Se a solicitação de posicionamento da sessão de jogo incluir dados de latência específicos da localização dos jogadores, Amazon GameLift Servers calcula a latência média do jogador em cada local e tenta colocar uma sessão de jogo em um local da frota com a menor média.
2. **Custo** — se uma solicitação não incluir dados de latência ou se várias frotas tiverem a mesma latência, Amazon GameLift Servers avaliará o custo de hospedagem de cada frota. O custo de hospedagem de uma frota varia de acordo com o tipo de frota (spot ou sob demanda), tipo de instância e localização.
3. **Destino** — Se várias frotas tiverem latência e custos iguais, Amazon GameLift Servers priorizará as frotas com base na ordem de destino, conforme listado na configuração da fila.
4. **Localização** — Para filas com frotas com vários locais, se todos os outros critérios forem iguais, Amazon GameLift Servers priorizará as localizações da frota com base na ordem alfabética.

Personalize como uma fila prioriza os posicionamentos das sessões de jogo

Você pode escolher personalizar como uma fila prioriza os critérios de posicionamento. A fila aplica a priorização personalizada a todas as solicitações de posicionamento de sessão de jogo que recebe.

Note

Se você criar uma configuração de prioridade personalizada e não incluir todos os quatro critérios, Amazon GameLift Servers anexará automaticamente os critérios ausentes na ordem padrão.

Para personalizar a configuração de prioridade de uma fila

Use o [Amazon GameLift Serversconsole](#) ou o AWS Command Line Interface (AWS CLI) para criar uma configuração de prioridade personalizada.

Console

No [Amazon GameLift Serversconsole](#), você pode personalizar as prioridades de uma fila ao criar uma nova fila ou atualizar uma fila existente. Selecione uma AWS região para trabalhar.

Abra a barra de navegação esquerda do console e escolha Filas. Na página Filas, selecione uma fila existente e escolha Editar.

1. Vá para a seção Prioridade de posicionamento da sessão de jogo. Arraste e solte cada critério de prioridade para criar o pedido desejado.
2. Vá para a seção Ordem de localização. Adicione todos os locais que você deseja priorizar. Essa lista é útil quando a fila tem frotas com vários locais. No mínimo, você deve especificar um local. Os locais que você especificar aqui são priorizados primeiro, seguidos por todos os outros locais nos destinos da fila.
3. Escolha Salvar alterações.

AWS CLI

Use o [update-game-session-queue](#) comando com a `--priority-configuration` opção de personalizar a ordem de prioridade de uma fila. Amazon GameLift Servers atualiza uma fila na sua AWS região padrão atual, ou você pode adicionar uma `--region` tag para especificar uma AWS região diferente.

O exemplo de solicitação a seguir adiciona ou atualiza a configuração de prioridade de uma fila especificada.

```
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" \  

```

Priorize as colocações de acordo com a latência do jogador

Se você quiser oferecer aos seus jogadores a melhor experiência possível e garantir o mínimo de latência, siga as etapas a seguir ao configurar seu sistema de posicionamento de sessões de jogo:

- Configure sua fila para priorizar a latência ao escolher onde colocar as sessões de jogo. Por padrão, a latência está no topo da lista de prioridades. Você também pode personalizar a configuração de prioridade da sua fila e escolher onde colocar a latência em ordem de prioridade.
- Configure políticas de latência do player para sua fila. As políticas de latência permitem que você defina limites rígidos sobre a quantidade de latência permitida na colocação de uma sessão de jogo. Se não Amazon GameLift Servers conseguir realizar uma sessão de jogo sem exceder os limites, a solicitação de colocação expirará e falhará. Você pode configurar uma única política de latência ou criar uma série de políticas que relaxem gradualmente o limite de latência ao longo do

tempo. Com uma série de políticas, você pode especificar limites de latência inicial muito baixos e ainda acomodar jogadores com latências mais altas após um pequeno atraso. Para obter detalhes sobre a criação de políticas de latência, consulte [Crie uma política de latência do jogador](#).

- Ao fazer solicitações de posicionamento de sessão de jogo (consulte [StartGameSessionPlacement](#)), inclua dados de latência de cada jogador. Os dados de latência do jogador incluem um valor para cada local possível em que uma sessão de jogo pode ser realizada. Por exemplo, para uma fila que coloca sessões de jogo em Regiões da AWS us-east-2 e ca-central-1, os dados de latência podem ter a seguinte aparência:

```
"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" }  
]
```

Para obter medições precisas de latência, use os Amazon GameLift Servers beacons de ping UDP. Esses endpoints permitem medir a latência real da rede UDP entre os dispositivos do player e cada um dos possíveis locais de hospedagem, resultando em decisões de posicionamento mais precisas do que o uso de pings ICMP. Para obter mais informações sobre o uso de beacons de ping UDP para medir a latência, consulte [Sinalizadores de ping UDP](#).

Priorize os posicionamentos por localização

Você pode configurar uma fila para fazer posicionamentos de sessões de jogo com base em uma lista priorizada de localizações geográficas. A localização é um dos critérios que determinam como uma fila escolhe onde colocar uma nova sessão de jogo. Por padrão, a localização é priorizada em quarto lugar, depois da latência, do custo e do destino.

Para o posicionamento da sessão de jogo, destino e localização têm significados um pouco diferentes:

- Destino se refere a uma frota específica e inclui todos os recursos de hospedagem da frota, onde quer que estejam implantados. Ao priorizar por destino, Amazon GameLift Servers pode fazer

um posicionamento em qualquer local da frota. As frotas gerenciadas em vários locais e as frotas Anywhere podem ter recursos de hospedagem implantados em um ou mais locais.

- A localização se refere a uma posição geográfica específica em que os recursos de hospedagem de uma frota são implantados. Uma frota pode ter vários locais, que podem incluir Regiões da AWS Zonas Locais ou locais personalizados (para uma frota em qualquer lugar). Uma frota gerenciada em um único local tem um local e é sempre um Região da AWS. Uma frota gerenciada em vários locais tem uma região de origem e pode ter locais remotos. Uma frota Anywhere tem um ou mais locais personalizados.

Ao priorizar os posicionamentos por local, Amazon GameLift Servers procura qualquer destino de fila que inclua o local prioritário e busca um recurso de hospedagem disponível. Se houver vários destinos com a localização prioritária, Amazon GameLift Servers passe para os próximos critérios de prioridade (custo, latência, destino).

Há várias maneiras de influenciar a forma como as localizações de uma fila são priorizadas.

- Configure como a fila lida com todas as solicitações de posicionamento da sessão do jogo:
 - Adicione uma configuração prioritária à fila. A configuração prioritária de uma fila inclui uma lista ordenada de locais. Você pode especificar um ou mais locais para priorizar. Essa lista não exclui nenhum local, ela simplesmente diz Amazon GameLift Servers onde procurar primeiro um recurso de hospedagem disponível. Um uso comum de uma lista ordenada de locais é quando você deseja canalizar a maior parte do tráfego para uma ou mais localizações geográficas específicas e usar locais adicionais como capacidade de backup. Adicione uma configuração prioritária ligando [UpdateGameSessionQueue](#).
 - Adicione uma configuração de filtro à fila. Uma configuração de filtro é uma lista de permissões para a fila. Ele diz Amazon GameLift Servers para ignorar todos os locais que não estão na lista ao procurar um recurso de hospedagem disponível. Há dois usos comuns para uma configuração de filtro. Primeiro, para frotas com vários locais, você pode usar um filtro para excluir alguns dos locais da frota. Em segundo lugar, talvez você queira proibir temporariamente as colocações em um determinado local; por exemplo, um local pode estar enfrentando problemas transitórios. Como você pode atualizar a configuração do filtro de uma fila a qualquer momento, você pode facilmente adicionar e remover locais conforme necessário. Adicione uma configuração de filtro chamando [UpdateGameSessionQueue](#).
- Use instruções especiais para solicitações individuais de colocação:

- Inclua uma lista de substituição prioritária em uma solicitação de colocação em uma sessão de jogo. Você pode fornecer uma lista alternativa de prioridades de locais com qualquer [StartGameSessionPlacement](#) solicitação. Essa lista substitui efetivamente a priorização configurada da fila para locais somente para aquela solicitação. Isso não afeta nenhuma outra solicitação. Esse recurso de substituição tem alguns requisitos:
 - Use uma lista de substituição somente com uma fila que tenha uma configuração de prioridade estabelecida LOCATION como primeira prioridade.
 - Não inclua dados de latência do jogador na mesma solicitação de posicionamento. Incluir dados de latência configura conflitos ao priorizar locais que não Amazon GameLift Servers podem ser resolvidos.
 - Decida como você Amazon GameLift Servers deseja proceder se ele não conseguir encontrar um recurso disponível na lista de substituição prioritária. Escolha entre voltar para os outros locais da fila ou limitar os posicionamentos à lista de substituição. Por padrão, Amazon GameLift Servers volta à tentativa de posicionamento em outros locais da fila.
 - Atualize a configuração do filtro da fila conforme necessário, como adicionar locais na lista de substituição. A lista de substituição não invalida a lista de filtros.

Crie uma política de latência do jogador

Se suas solicitações de posicionamento incluírem dados de latência do jogador Amazon GameLift Servers, encontrará sessões de jogo em locais com a menor latência média para todos os jogadores. A colocação de sessões de jogo com base na latência média do jogador Amazon GameLift Servers impede a colocação da maioria dos jogadores em jogos com alta latência. No entanto, Amazon GameLift Servers ainda coloca jogadores com latência extrema. Para acomodar esses jogadores, crie políticas de latência do jogador.

Uma política de latência do jogador Amazon GameLift Servers impede colocar uma sessão de jogo solicitada em qualquer lugar em que os jogadores da solicitação experimentem latência acima do valor máximo. As políticas de latência do jogador também podem Amazon GameLift Servers impedir a correspondência de solicitações de sessão de jogo com jogadores de maior latência.

Tip

Para gerenciar regras específicas de latência, como exigir latência semelhante em todos os jogadores de um grupo, você pode usar [Amazon GameLift Servers FlexMatch](#) para criar regras de matchmaking baseadas em latência.

Por exemplo, considere essa fila com um tempo limite de cinco minutos e as seguintes políticas de latência do jogador:

1. Passe 120 segundos procurando um local onde todas as latências do jogador sejam inferiores a 50 milissegundos.
2. Passe 120 segundos procurando um local onde todas as latências do jogador sejam inferiores a 100 milissegundos.
3. Passe o tempo restante da fila até o tempo limite procurando um local onde todas as latências do jogador sejam inferiores a 200 milissegundos.

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

Avaliar métricas de fila

Use métricas para avaliar o desempenho das suas filas. Você pode visualizar métricas relacionadas às filas no [Amazon GameLift Servers console](#) ou na Amazon CloudWatch. Para obter uma lista e descrições de métricas de fila, consulte [Métricas do Amazon GameLift Servers para filas](#).

As métricas de fila podem fornecer informações sobre o seguinte:

- Desempenho geral da fila: as métricas da fila indicam com que sucesso uma fila responde às solicitações de posicionamento. Essas métricas também podem ajudá-lo a identificar quando e por que os posicionamentos falham. Para filas com frotas escaladas manualmente, as métricas `AverageWaitTime` e `QueueDepth` podem indicar quando você deve ajustar a capacidade de uma fila.
- FleetIQ desempenho do algoritmo — Para solicitações de posicionamento usando o FleetIQ algoritmo, as métricas mostram com que frequência o algoritmo encontra o posicionamento ideal da sessão de jogo. O posicionamento pode priorizar o uso de recursos com a menor latência do jogador ou recursos com o menor custo. Também existem métricas de erro que identificam motivos comuns Amazon GameLift Servers não consigo encontrar um posicionamento ideal. Para obter mais informações sobre métricas, consulte [Monitore Amazon GameLift Servers com a Amazon CloudWatch](#).
- Posicionamentos específicos de locais: para filas de vários locais, as métricas mostram posicionamentos bem-sucedidos por local. Para filas que usam o FleetIQ algoritmo, esses dados fornecem informações úteis sobre onde a atividade do jogador ocorre.

Ao avaliar métricas para FleetIQ desempenho do algoritmo, considere as seguintes dicas:

- Para rastrear a taxa de localização ideal da fila, use a `PlacementsSucceeded` métrica em combinação com a FleetIQ métricas para menor latência e menor preço.
- Para aumentar a taxa da fila de descobertas de um posicionamento ideal, analise as seguintes métricas de erro:
 - Se `FirstChoiceOutOfCapacity` for alto, ajuste o escalonamento da capacidade para as frotas da fila.
 - Se a métrica `FirstChoiceNotViable` de erro for alta, veja suas frotas de instâncias Spot. As frotas de instâncias spot são consideradas inviáveis quando a taxa de interrupção de um determinado tipo de instância é muito alta. Para resolver esse problema, altere a fila de modo que ela use as frotas de instâncias spot com diferentes tipos de instância. Recomendamos que você inclua frotas de instâncias spot com diferentes tipos de instância em cada local.

Criar uma fila para instâncias spot

Você pode aproveitar uma economia significativa nos custos de hospedagem usando frotas Spot. Consulte mais detalhes em [Instâncias sob demanda versus instâncias spot](#). Para adicionar frotas

spot à sua solução de hospedagem, você precisa configurar uma fila de sessões de jogo com uma combinação de frotas spot e frotas sob demanda. Amazon GameLift Servers usa uma fila durante o processo de colocação da sessão de jogo para pesquisar em várias frotas e encontrar os melhores anfitriões disponíveis para novas sessões de jogo. Este tópico fornece orientação sobre como começar a usar frotas Spot.

Você está usando FlexMatch para matchmaking? Você pode usar as etapas a seguir para adicionar frotas Spot às suas filas de sessões de jogo existentes para colocações de matchmaking.

1. Determine os destinos da fila de sessões de jogo.

Gerenciar o posicionamento da sessão de jogo com uma fila é sempre uma melhor prática necessária ao usar instâncias spot. Como as instâncias spot nem sempre estão disponíveis quando necessário, é preciso projetar uma fila resiliente que inclua frotas spot e frotas sob demanda para oferecer capacidade de backup. Você pode manter as frotas sob demanda em escala vertical reduzida até que elas sejam necessárias. Para projetar a fila, considere o seguinte:

- Localizações — Se possível, suas frotas spot e sob demanda devem estar na mesma região que os jogadores. Posicione os recursos spot e os recursos sob demanda em cada local que você deseja oferecer suporte. As frotas de vários locais oferecem suporte a instâncias spot e sob demanda.
- Tipos de instância — Considere os requisitos de hardware do seu servidor de jogos e a disponibilidade de instâncias nos locais que você escolher.

Para experimentar uma fila que otimize a disponibilidade e a resiliência do spot, consulte [Tutorial: Crie uma Amazon GameLift Servers fila com instâncias spot](#). Para ver as práticas recomendadas de design de spot, consulte [Práticas recomendadas para filas de sessões de Amazon GameLift Servers jogos](#).

2. Crie as frotas para a sua fila otimizada para spot.

Com base no design da fila, crie frotas para implantar os servidores de jogos nos locais e tipos de instância desejados. Consulte [Crie uma EC2 frota Amazon GameLift Servers gerenciada](#) para receber ajuda sobre como criar e configurar novas frotas.

3. Criar uma fila de sessões de jogos.

Adicione os destinos da frota, configure o processo de posicionamento da sessão de jogo e defina as prioridades de posicionamento. Consulte [Criar uma fila de sessões de jogo](#) para receber ajuda sobre como criar e configurar a nova fila.

4. Atualize o serviço de cliente do jogo para usar a fila.

Quando o cliente do jogo usa uma fila para solicitar recursos, a fila evita recursos com alta chance de interrupção e seleciona o local que corresponde às suas prioridades definidas. Para receber ajuda sobre como implantar posicionamentos de sessões de jogo no seu jogo, consulte [Criar sessões de jogos](#).

5. Atualize o servidor de jogos lidar com interrupções spot.

AWS pode interromper as Instâncias Spot com uma notificação de 2 minutos, quando precisar recuperar a capacidade. Configure o servidor de jogos para lidar com interrupções e minimizar o impacto do jogador.

Antes de AWS recuperar uma Instância Spot, ela envia uma notificação de encerramento. Amazon GameLift Servers passa a notificação para todos os processos do servidor afetados invocando o Amazon GameLift Servers Função de retorno de chamada do SDK do servidor. `onProcessTerminate()` Implemente esse retorno de chamada para encerrar a sessão de jogo ou mover a sessão de jogo e os jogadores para uma nova instância. Consulte [Responder a uma notificação de desligamento do processo do servidor](#) para receber ajuda sobre como implementar `onProcessTerminate()`.

 Note

AWS faz todos os esforços para fornecer a notificação antes de recuperar uma instância, mas é possível que AWS recupere a Instância Spot antes que o aviso chegue. Prepare o servidor de jogos para lidar com interrupções inesperadas.

6. Revise o desempenho das frotas spot e das filas.

Visualizar Amazon GameLift Servers métricas no Amazon GameLift Servers console ou com a Amazon CloudWatch para avaliar o desempenho. Para obter mais informações sobre Amazon GameLift Servers métricas, veja [Monitore Amazon GameLift Servers com a Amazon CloudWatch](#). As principais métricas incluem:

- Taxa de interrupção: use as métricas `InstanceInterruptions` e `GameSessionInterruptions` para acompanhar a quantidade e a frequência das

interrupções relacionadas a instâncias spot e sessões de jogos. As sessões de jogo que são recuperadas por AWS têm um status de TERMINATED e um motivo de status de INTERRUPTED

- Eficácia da fila: acompanhe as taxas de sucesso de posicionamento, tempo médio de espera e profundidade da fila, para verificar se o uso de frotas spot não tem impacto no desempenho da fila.
- Uso da frota: monitore dados sobre instâncias, sessões de jogos e sessões de jogadores. O uso de frotas sob demanda pode ser um indicador de que as filas estão evitando posicionamentos em suas frotas spot para evitar interrupções.

Melhores práticas para filas com frotas Spot

Se sua fila incluir frotas Spot, configure uma fila resiliente. Isso aproveita a economia de custos com as frotas Spot e, ao mesmo tempo, minimiza o efeito das interrupções nas sessões de jogo. Para obter ajuda sobre como criar frotas e filas de sessões de jogo corretamente para uso com frotas Spot, consulte [Tutorial: Crie uma Amazon GameLift Servers fila com instâncias spot](#). Para obter mais informações sobre instâncias spot, consulte [Criar uma fila para instâncias spot](#).

Além das melhores práticas gerais na seção anterior, considere estas melhores práticas específicas do SPOT:

- Crie pelo menos uma frota sob demanda em cada local. As frotas sob demanda fornecem servidores de jogos de backup para seus jogadores. Você poderá reduzir a escala de suas frotas de backup até que sejam necessárias e usar o ajuste de escala automático para aumentar a capacidade sob demanda quando as frotas spot não estiverem disponíveis.
- Selecione diferentes tipos de instância em várias frotas Spot em um local. Se um tipo de Instância Spot ficar temporariamente indisponível, a interrupção afetará somente uma frota Spot no local. A melhor prática é escolher tipos de instância amplamente disponíveis e usar tipos de instância da mesma família (por exemplo, m5.large, m5.xlarge, m5.2xlarge). Usar a [Amazon GameLift Servers console](#) para visualizar dados históricos de preços para tipos de instância.

Configurar notificações de eventos para o posicionamento de sessões de jogo

Você pode usar as notificações de eventos para monitorar o status de solicitações individuais de posicionamento. Recomendamos configurar notificações de eventos para todos os jogos com alto volume de atividades de posicionamento.

Há duas opções para a configuração de notificações de eventos.

- Tenho Amazon GameLift Servers publique notificações de eventos em um tópico do Amazon Simple Notification Service (Amazon SNS) usando uma fila.
- Use EventBridge eventos da Amazon publicados automaticamente e seu conjunto de ferramentas para gerenciar eventos.

Para obter uma lista de eventos de posicionamento de sessões de jogo emitidos por Amazon GameLift Servers, consulte [Eventos de posicionamento de sessões de jogo](#).

Configurar um tópico do SNS

Para Amazon GameLift Servers para publicar todos os eventos gerados por uma fila de sessão de jogo em um tópico, defina o campo de destino da notificação como um tópico.

Para configurar um tópico do SNS para Amazon GameLift Servers notificação de evento

1. [Faça login no AWS Management Console e abra o console do Amazon SNS em https://console.aws.amazon.com/sns/v3/home](https://console.aws.amazon.com/sns/v3/home).
2. Na página Topics do SNS, escolha Create topic (Criar tópico) e siga as instruções para criá-lo.
3. Em Access Policy (Política de acesso), faça o seguinte:
 - a. Escolha o método Advanced (Avançado).
 - b. Adicione a seguinte seção em negrito do objeto JSON à política existente.

```
{
  "Version": "2008-10-17",
  "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:your_region:your_account: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:your_region:your_account:your_topic_name",
    "Condition": {
      "ArnLike": {
        "aws:SourceArn":
          "arn:aws:gamelift:your_region:your_account:gamesessionqueue/your_queue_name"
      }
    }
  }
]
}

```

- c. (Opcional) Adicione mais controle de acesso ao tópico incluindo condições à política de recursos.

4. Escolha Create topic (Criar tópico).

5. Depois de criar seu tópico do SNS, adicione-o às filas durante a criação das filas ou edite uma fila existente para adicioná-lo.

Configurar um tópico do SNS com a criptografia do lado do servidor

Com a criptografia do lado do servidor (SSE), é possível armazenar dados confidenciais em tópicos criptografados. A SSE protege o conteúdo de mensagens nos tópicos do Amazon SNS usando chaves gerenciadas no AWS Key Management Service (AWS KMS). Para obter mais informações sobre criptografia no lado do servidor com o Amazon SNS, consulte [Criptografia em repouso](#) no Guia do desenvolvedor do Amazon Simple Notification Service.

Para configurar um tópico do SNS com a criptografia do lado do servidor, consulte os tópicos a seguir:

- [Criar chaves](#) no Guia do desenvolvedor do AWS Key Management Service
- [Habilitar a SSE para um tópico](#) no Guia do desenvolvedor do Amazon Simple Notification Service

Ao criar a chave do KMS, use a seguinte política de chave do 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"
    }
  }
}
```

Configurar EventBridge

Amazon GameLift Servers publica automaticamente todos os eventos de posicionamento da sessão de jogo em EventBridge. Com EventBridge você pode configurar regras para que os eventos sejam roteados aos alvos para processamento. Por exemplo, você pode definir uma regra para rotear o evento `PlacementFulfilled` para uma AWS Lambda função que gerencia tarefas que precedem a conexão com uma sessão de jogo. Para obter mais informações sobre EventBridge, consulte [O que é a Amazon EventBridge?](#) no Guia do EventBridge usuário da Amazon.

A seguir estão alguns exemplos de EventBridge regras para usar com Amazon GameLift Servers filas:

Combina eventos de todos Amazon GameLift Servers filas

```
{
  "source": [
    "aws.gamelift"
  ],
  "detail-type": [
    "GameLift Queue Placement Event"
  ]
}
```

Faz correspondência com eventos de uma fila específica

```
{
  "source": [
    "aws.gamelift"
  ],
  "detail-type": [
    "GameLift Queue Placement Event"
  ],
  "resources": [
    "arn:aws:gamelift:your_region:your_account:gamesessionqueue/your_queue_name"
  ]
}
```

Tutorial: Crie uma Amazon GameLift Servers fila com instâncias spot

Introdução

Este tutorial descreve como configurar o posicionamento de sessões de jogo para jogos implantados em frotas Spot de baixo custo. As frotas spot requerem etapas adicionais para manter a disponibilidade contínua do servidor de jogos para os jogadores.

Público-alvo

Este tutorial é para desenvolvedores de jogos que desejam usar frotas Spot para hospedar servidores de jogos personalizados ou Amazon GameLift ServersRealtime.

O que você aprenderá

- Defina o grupo de jogadores que a fila de sessões de jogo atende.
- Crie uma infraestrutura de frota para dar suporte ao escopo da fila de sessões de jogo.
- Atribua um alias a cada frota para abstrair o ID da frota.
- Crie uma fila, adicione frotas e priorize onde Amazon GameLift Servers as sessões de jogo são realizadas.
- Adicione políticas de latência do jogador para ajudar a minimizar os problemas de latência.

Pré-requisitos

Antes de criar frotas e filas para o posicionamento das sessões de jogo, execute as seguintes tarefas:

- Consulte [Como funciona o Amazon GameLift Servers](#).
- [Integre seu servidor de jogos com Amazon GameLift Servers](#) o.
- [Carregue a compilação ou o Realtime script do seu servidor de jogo](#) paraAmazon GameLift Servers.
- [Planeje a configuração da frota](#).

Etapa 1: Definir o escopo da fila

Neste tutorial, criamos uma fila para um jogo que tem uma variação de compilação de servidor de jogos. No lançamento, lançaremos o jogo em dois locais: Ásia-Pacífico (Seul) e Ásia-Pacífico

(Singapura). Como esses locais estão próximos um do outro, a latência não é um problema para nossos jogadores.

Neste exemplo, há um segmento de jogadores, o que significa que criamos uma fila. No futuro, quando lançarmos o jogo na América do Norte, poderemos criar uma segunda fila voltada para jogadores norte-americanos.

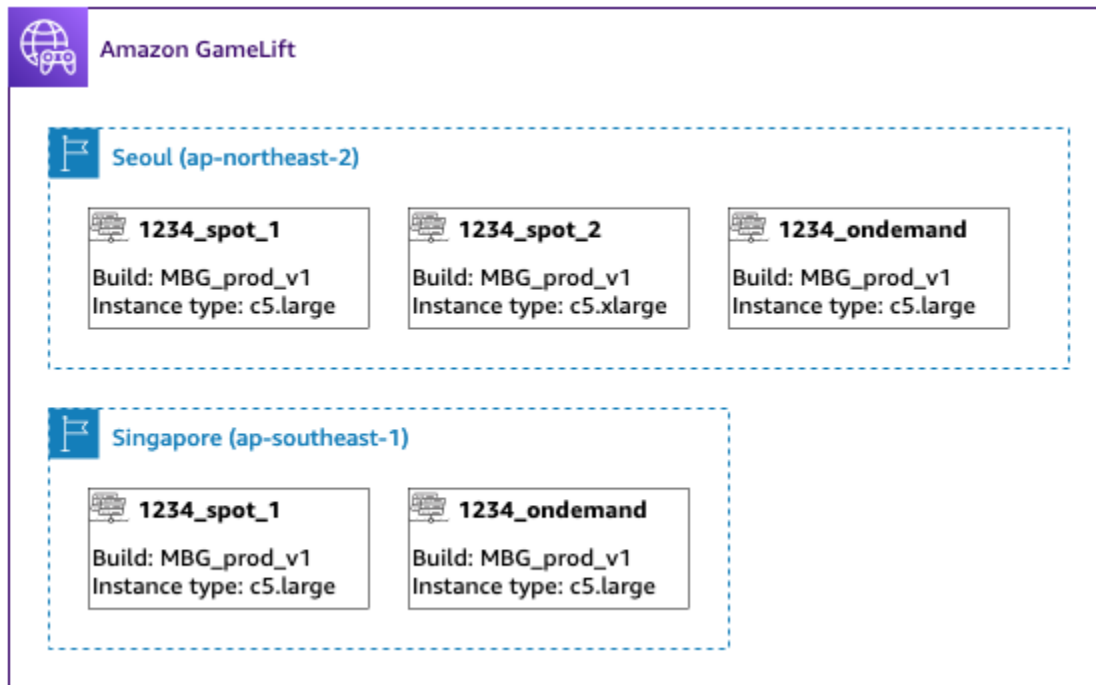
Para obter mais informações, consulte [Definir o escopo de uma fila](#).

Etapa 2: Criar uma infraestrutura de frota spot

Crie frotas em locais e com compilações de servidores de jogos ou scripts que se encaixem no escopo que você definiu em [Etapa 1: Definir o escopo da fila](#).

Neste tutorial, criamos uma infraestrutura de dois locais com, pelo menos, uma frota spot e uma frota sob demanda em cada local. Cada frota implanta a mesma versão de servidor de jogos. Além disso, prevemos que o tráfego de jogadores será mais intenso na localização de Seul e, portanto, adicionamos mais frotas spot lá.

O diagrama a seguir mostra o exemplo da infraestrutura da frota spot, com três frotas na localização ap-northeast-2 (Seul) e duas frotas na localização ap-southeast-1 (Singapura). Todas as instâncias em ambas as frotas estão usando a compilação MBG_Prod_v1. A frota em ap-northeast-2 contém as seguintes configurações de frota: frota 1234_spot_1 com um tipo de instância de c5.large, frota 1234_spot_2 com um tipo de instância de c5.xlarge e frota 1234_ondemand com um tipo de instância de c5.large. A frota em ap-southeast-1 contém as seguintes configurações de frota: frota 1234_spot_1 com um tipo de instância de c5.large e frota 1234_ondemand com um tipo de instância de c5.large.

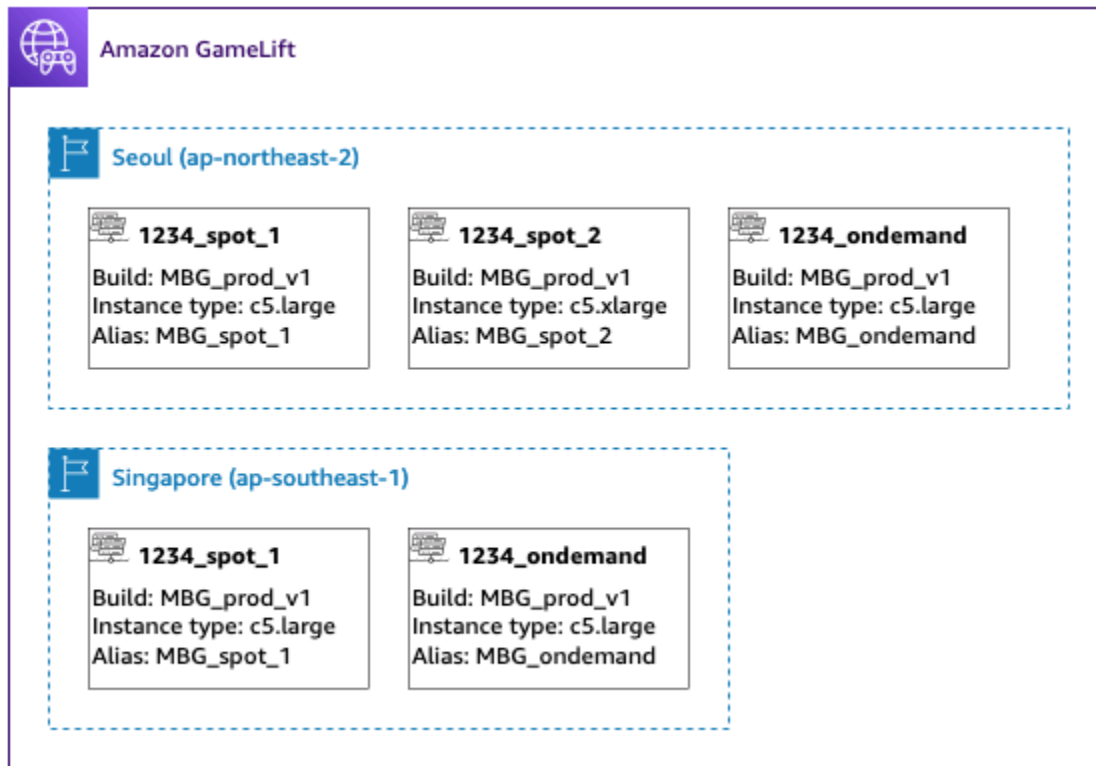


Etapa 3: Atribuir aliases para cada frota

Crie um novo alias para cada frota na sua infraestrutura. Os aliases abstraem as identidades da frota, tornando eficiente a substituição periódica da frota. Para obter mais informações sobre como criar aliases, consulte [Crie um Amazon GameLift Servers alias](#).

Nossa infraestrutura de frota tem cinco frotas e, portanto, criamos cinco aliases usando a estratégia de roteamento. Precisamos de três aliases no local Ásia-Pacífico (Seul) e dois aliases no local Ásia-Pacífico (Singapura).

O diagrama a seguir mostra a infraestrutura da frota spot descrita na etapa dois com aliases adicionados a cada frota. A frota 1234_spot_1 tem o alias MBG_spot_1, a frota 1234_spot_2 tem o alias MBG_spot_2 e a frota 1234_ondemand tem o alias MBG_ondemand.



Para obter mais informações, consulte [Criar uma fila de vários locais](#).

Etapa 4: Criar uma fila com destinos

Crie a fila da sessão de jogo e adicione os destinos da frota. Para obter mais informações sobre como criar uma fila, consulte [Criar uma fila de sessões de jogo](#).

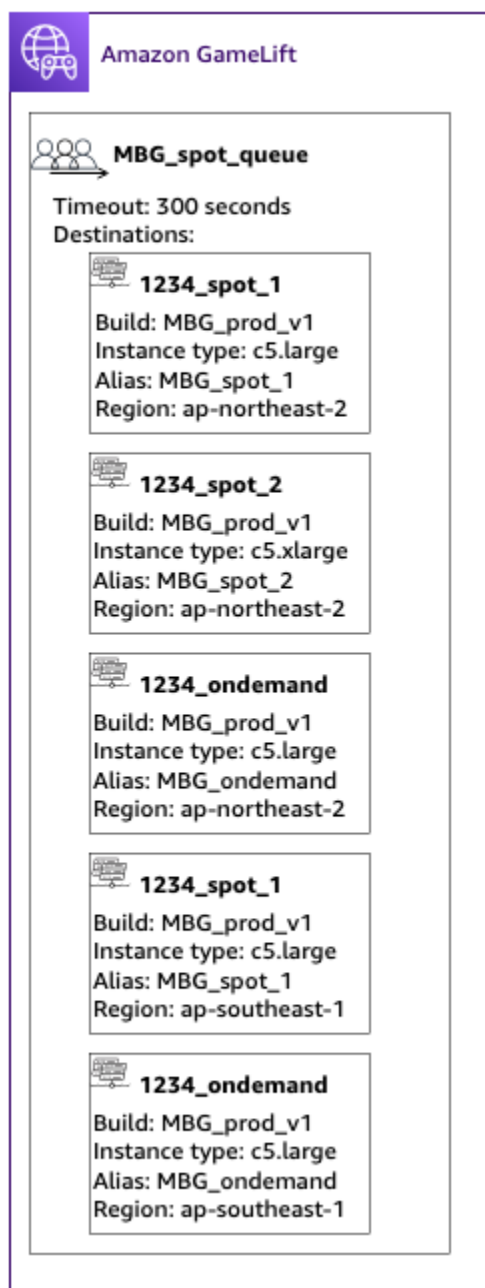
Ao criar sua fila:

- O valor do tempo limite padrão é 10 minutos. Posteriormente, é possível testar como o tempo limite da fila afeta o tempo de espera dos jogadores para entrar nos jogos.
- Por enquanto, pule a seção sobre políticas de latência do jogador. Abordaremos isso na próxima etapa.
- Priorize as frotas na sua fila. Ao trabalhar com frotas spot, recomendamos uma das seguintes abordagens:
 - Se sua infraestrutura usa um local principal com frotas em um segundo local para backup, priorize as frotas primeiro por local e, depois, por tipo de frota.
 - Se sua infraestrutura usa vários locais igualmente, priorize as frotas por tipo de frota, colocando as frotas spot no topo da fila.

Neste tutorial, criamos uma nova fila com o nome **MBG_spot_queue** e adicionamos os aliases das nossas cinco frotas. Em seguida, priorizamos os posicionamentos primeiro por local e, depois, por tipo de frota.

Com base nessa configuração, essa fila sempre tenta posicionar novas sessões de jogo em uma frota spot em Seul. Quando essas frotas estão cheias, a fila usa a capacidade disponível na frota sob demanda de Seul como reserva. Se todas as três frotas de Seul não estiverem disponíveis, Amazon GameLift Servers coloca sessões de jogos nas frotas de Cingapura.

O diagrama a seguir mostra uma fila com um tempo limite de 300 segundos e destinos priorizados. Os destinos estão na seguinte ordem: 1234_spot_1 em ap-northeast-2, 1234_spot_2 em ap-northeast-2, 1234_ondemand em ap-northeast-2, 1234_spot_1 em ap-southeast-1 e 1234_ondemand em ap-southeast-1.



Etapa 5: Adicionar limites de latência à fila

Nosso jogo inclui informações de latência nas solicitações de posicionamento da sessão de jogo. Também temos um atributo de festa de jogadores que cria uma sessão de jogo para um grupo de jogadores. Podemos fazer com que os jogadores esperem um pouco mais para entrar nos jogos com a experiência de jogo ideal. Nossos testes de jogos mostram as seguintes observações:

- Uma latência abaixo de 50 milissegundos é ideal.

- O jogo não pode ser jogado em latências superiores a 250 milissegundos.
- Os jogadores ficam impacientes por volta de um minuto.

Para nossa fila, com um tempo limite de 300 segundos, adicionamos declarações de política que limitam a latência permitida. As declarações de política permitem gradualmente maiores valores de latência de até 250 milissegundos.

Com essa política, nossa fila procura posicionamentos com latência ideal (menos de 50 milissegundos) no primeiro minuto e, depois, relaxa o limite. A fila não faz posicionamentos em que a latência do jogador seja de 250 milissegundos ou mais.

O diagrama a seguir mostra a fila da etapa quatro com as políticas de latência do jogador adicionadas. As políticas de latência do jogador estabelecem o seguinte: imposição do limite de 50 ms por 60 s, imposição do limite de 125 ms por 30 s e imposição do limite de 250 ms até o tempo limite.

 Amazon GameLift

 **MBG_spot_queue**

Timeout: 300 seconds

Destinations:

 **1234_spot_1**
Build: MBG_prod_v1
Instance type: c5.large
Alias: MBG_spot_1
Region: ap-northeast-2

 **1234_spot_2**
Build: MBG_prod_v1
Instance type: c5.xlarge
Alias: MBG_spot_2
Region: ap-northeast-2

 **1234_ondemand**
Build: MBG_prod_v1
Instance type: c5.large
Alias: MBG_ondemand
Region: ap-northeast-2

 **1234_spot_1**
Build: MBG_prod_v1
Instance type: c5.large
Alias: MBG_spot_1
Region: ap-southeast-1

 **1234_ondemand**
Build: MBG_prod_v1
Instance type: c5.large
Alias: MBG_ondemand
Region: ap-southeast-1

Latency policies:

- Enforce 50ms limit for 60s
- Enforce 125ms limit for 30s
- Enforce 250ms limit until timeout

Resumo

Parabéns! Aqui estão as coisas que você realizou:

- Você tem uma fila de sessões de jogo com escopo definido para um segmento da sua população de jogadores.
- Sua fila usa frotas spot de forma eficaz e é resiliente quando ocorrem interrupções do spot.
- Sua fila prioriza as frotas para uma melhor experiência do jogador.
- A fila tem limites de latência para proteger os jogadores de experiências de jogo ruins.

Agora você pode usar a fila para colocar sessões de jogo para os jogadores que ela atende. Ao fazer solicitações de posicionamento de sessão de jogo para esses jogadores, faça referência ao nome dessa fila de sessões de jogo na solicitação. Para obter mais informações sobre como fazer solicitações de posicionamento de sessão de jogo, consulte [Criar sessões de jogos](#) ou [Integrando um cliente de jogo para Amazon GameLift ServersRealtime](#).

Próximas etapas:

- [Projete sua própria fila](#).
- [Crie sua fila](#).
- [Use uma fila com seu cliente de jogo](#).

Escalando a capacidade de hospedagem de jogos com Amazon GameLift Servers

A capacidade de hospedagem, medida em instâncias, representa o número de sessões de jogo que Amazon GameLift Servers podem hospedar simultaneamente e o número de jogadores simultâneos que essas sessões de jogo podem acomodar. Uma das tarefas mais desafiadoras da hospedagem de jogos é escalar a capacidade para atender à demanda dos jogadores sem desperdiçar dinheiro com recursos que não sejam necessários. Para obter mais informações, consulte [Escalar a capacidade de frota](#).

A capacidade é ajustada no nível de localização da frota. Todas as frotas têm pelo menos um local: a AWS região de origem da frota. Ao visualizar ou escalar a capacidade, as informações são listadas por local, incluindo a região de origem da frota e quaisquer locais remotos adicionais.

Defina manualmente o número de instâncias a serem mantidas ou configure o ajuste de escala automático para ajustar dinamicamente a capacidade à medida que a demanda dos jogadores muda. É recomendável começar ativando a opção de ajuste de escala automático baseado em destino. O objetivo do ajuste de escala automático com base em origem é manter recursos de hospedagem suficientes para acomodar os jogadores atuais, além de um pouco mais para lidar com picos inesperados na demanda dos jogadores. Para a maioria dos jogos, o ajuste de escala automático com base em origem, oferece uma solução de escalabilidade altamente eficaz.

Você pode realizar a maioria das atividades de escalonamento de frota usando o Amazon GameLift Servers console. Você também pode usar um AWS SDK ou o AWS Command Line Interface (AWS CLI) com a [API de serviço para Amazon GameLift Servers](#).

Tópicos

- [Para gerenciar a capacidade da frota no console](#)
- [Defina limites Amazon GameLift Servers de capacidade](#)
- [Defina manualmente a capacidade de uma Amazon GameLift Servers frota](#)
- [Capacidade de frota de escala automática com Amazon GameLift Servers](#)
- [Escale frotas de Amazon GameLift Servers contêineres](#)

Para gerenciar a capacidade da frota no console

1. Abra o [console de Amazon GameLift Servers](#).
2. No painel de navegação, selecione Hospedagem, Frotas.
3. Na página Frotas, escolha o nome de uma frota ativa para abrir a página de detalhes da frota.
4. Escolha a guia Escalabilidade. Neste guia, você poderá:
 - Veja métricas históricas de escalabilidade para toda a frota.
 - Visualize e atualize as configurações de capacidade para cada localização da frota, incluindo limites de escala e configurações de capacidade atuais.
 - Atualize o ajuste de escala automático com base em destino, visualize as políticas de ajuste de escala automático com base em regras aplicadas a toda a frota e suspenda a atividade de ajuste de escala automático em cada local.

Defina limites Amazon GameLift Servers de capacidade

Ao escalar a capacidade de hospedagem para um local de Amazon GameLift Servers frota, manualmente ou por escalonamento automático, considere os limites de escalabilidade do local. Todas as localizações da frota têm um limite mínimo e máximo que definem o alcance permitido para a capacidade do local. Por padrão, os limites nos locais da frota têm um mínimo de 0 instâncias e um máximo de 1 instância. Antes de escalar a localização de uma frota, ajuste os limites.

Se você estiver usando o escalonamento automático, o limite máximo permite ampliar Amazon GameLift Servers a localização de uma frota para atender à demanda dos jogadores, mas evita custos excessivos de hospedagem, como durante um ataque de DDOS. Configure um [CloudWatch alarme da Amazon](#) para notificá-lo quando a capacidade se aproximar do limite máximo, para que você possa avaliar a situação e ajustar manualmente conforme necessário. (Você também pode [criar um alarme de cobrança](#) para monitorar AWS os custos.) O limite mínimo é útil para manter a disponibilidade da hospedagem, mesmo quando a demanda dos jogadores é baixa.

Você pode definir limites de capacidade para as localizações de uma frota no [Amazon GameLift Serversconsole](#) ou usando o AWS Command Line Interface (AWS CLI).

Como definir limites de capacidade

Console

1. Abra o [console de Amazon GameLift Servers](#).
2. No painel de navegação, selecione Hospedagem, Frotas.
3. Na página Frotas, escolha o nome de uma frota ativa para abrir a página de detalhes da frota.
4. Na guia Escalabilidade, em Capacidade de escalabilidade, selecione um local da frota e selecione Editar.
5. Na caixa de diálogo Editar capacidade de escalabilidade, defina as contagens de instâncias para Tamanho mínimo, Instâncias desejadas e Tamanho máximo.
6. Selecione a opção Confirmar.

AWS CLI

1. Verifique as configurações atuais de capacidade. Em uma janela de linha de comando, use o [describe-fleet-location-capacity](#) comando com o ID da frota e o local para os quais você deseja alterar a capacidade. Esse comando retorna um [FleetCapacity](#) objeto que inclui as configurações atuais de capacidade do local. Determine se a nova instância limite poderá acomodar a configuração atual de instâncias desejadas.

```
aws gamelift describe-fleet-location-capacity \
  --fleet-id <fleet identifier> \
  --location <location name>
```

2. Atualize as configurações de limite. Em uma janela de linha de comando, use o [update-fleet-capacity](#) comando com os seguintes parâmetros. Você pode ajustar os limites de instâncias desejadas e contagem de instâncias com o mesmo 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>
```

Exemplo:

```
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 sua solicitação for bem-sucedida, Amazon GameLift Servers retornará o ID da frota. Se o novo `min-size` valor `max-size` ou o valor entrarem em conflito com a `desired-instances` configuração atual, Amazon GameLift Servers retornará um erro.

Defina manualmente a capacidade de uma Amazon GameLift Servers frota

Quando você cria uma nova frota, define Amazon GameLift Servers automaticamente as instâncias desejadas para uma instância em cada localização da frota. Em seguida, Amazon GameLift Servers implanta uma nova instância em cada local. Para alterar a capacidade da frota, adicione uma política de ajuste de escala automático com base em destino ou defina manualmente o número de instâncias que deseja para um local. Para obter mais informações, consulte [Escalar a capacidade de frota](#).

Definir a capacidade de uma frota manualmente pode ser útil ao não precisar de ajuste de escala automático ou quando precisar manter a capacidade em um nível específico. A configuração manual da capacidade só funciona caso não esteja usando uma política de ajuste de escala automático com base em destino. Caso tenha uma política de ajuste de escala automático com base em destino, ela redefinirá imediatamente a capacidade desejada com base em suas próprias regras de escalabilidade.

Você pode definir manualmente a capacidade no Amazon GameLift Servers console ou usando o AWS Command Line Interface (AWS CLI). O status da frota deve ser ativo.

Suspender o ajuste de escala automático

Suspenda todas as atividades de ajuste de escala automático para cada localização da frota. Com a suspensão do ajuste de escala automático, o número desejado de instâncias no local da frota permanece o mesmo, a menos que seja alterado manualmente. Ao suspender o ajuste de escala

automático de um local, isso afeta as políticas atuais da frota e quaisquer políticas que possa definida no futuro.

Como definir manualmente capacidade de frota

Console

1. Abra o [console de Amazon GameLift Servers](#).
2. No painel de navegação, selecione Hospedagem, Frotas.
3. Na página Frotas, escolha o nome de uma frota ativa para abrir a página de detalhes da frota.
4. Na guia Escalabilidade, em Locais de ajuste de escala automático suspensos, selecione cada local para o qual você deseja suspender o ajuste de escala automático e, em seguida, escolha Suspend.
5. Em Capacidade de escalabilidade, selecione um local a ser definido manualmente e escolha Editar.
6. Na caixa de diálogo Editar capacidade de escalabilidade, defina seu valor preferido para as Instâncias desejadas e escolha Confirmar. Isso informa Amazon GameLift Servers o número de instâncias a serem mantidas em um estado ativo, prontas para hospedar sessões de jogo.

Amazon GameLift Servers responde às mudanças implantando instâncias adicionais ou encerrando instâncias desnecessárias. Ao Amazon GameLift Servers concluir esse processo, o número de instâncias ativas no local muda para corresponder ao valor atualizado das instâncias desejadas. Esse processo pode demorar um pouco.

AWS CLI

1. Verifique as configurações atuais de capacidade. Em uma janela de linha de comando, use o [describe-fleet-location-capacity](#) comando com o ID da frota e o local para os quais você deseja alterar a capacidade. Esse comando retorna um [FleetCapacity](#) objeto que inclui as configurações atuais de capacidade do local. Determine se os limites de instância podem acomodar a nova configuração desejada de instâncias.

```
aws gamelift describe-fleet-location-capacity \
  --fleet-id <fleet identifier> \
  --location <location name>
```

2. Atualizar a capacidade desejada. Use o [update-fleet-capacity](#) comando com o ID da frota, a localização e um novo valor para as instâncias desejadas. Se esse valor está fora do intervalo do limite atual, você poderá ajustar os valores limite no mesmo 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]
```

Exemplo:

```
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 sua solicitação for bem-sucedida, Amazon GameLift Servers retornará o ID da frota. Se a nova configuração de instâncias desejada estiver fora dos limites mínimo e máximo, Amazon GameLift Servers retornará um erro.

Capacidade de frota de escala automática com Amazon GameLift Servers

Use o auto-scaling em Amazon GameLift Servers para escalar dinamicamente a capacidade da sua frota em resposta à atividade do servidor do jogo. À medida que os jogadores chegam e iniciam as sessões de jogos, o ajuste de escala automático pode adicionar mais instâncias; à medida que a demanda dos jogadores diminui, o ajuste de escala automático pode finalizar instâncias desnecessárias. O ajuste de escala automático é uma forma eficiente para minimizar os recursos de hospedagem e custos, enquanto ainda fornece uma experiência de jogador suave e rápida.

Para usar o escalonamento automático, você cria políticas de escalabilidade que informam Amazon GameLift Servers quando aumentar ou diminuir a escala. Há dois tipos de políticas de escalabilidade: com base no destino e com base nas regras. A abordagem baseada em destino (rastreamento de destino) é uma solução completa. Nós o recomendamos como a opção mais simples e eficaz.

Políticas de escalabilidade com base em regras, que exigem que você defina cada aspecto do processo de realizar um ajuste de escala automático, que é útil para lidar com questões específicas. Esta solução funciona melhor como um suplemento para o ajuste de escala automático com base em destino.

Você pode gerenciar o escalonamento automático baseado em metas usando o Amazon GameLift Servers console, o AWS Command Line Interface (AWS CLI) ou um AWS SDK. Você pode gerenciar o escalonamento automático baseado em regras usando somente o SDK AWS CLI ou um AWS SDK, embora possa visualizar as políticas de escalabilidade com base em regras no console.

Tópicos

- [Ajuste de escala automático baseado em destino](#)
- [Ajuste de escala automático com políticas com base nas regras](#)

Ajuste de escala automático baseado em destino

Escalonamento automático baseado em metas para Amazon GameLift Servers ajusta os níveis de capacidade com base na métrica `PercentAvailableGameSessions` da frota. Esta métrica representa o buffer disponível da frota para aumentos repentinos na demanda do jogador.

O principal motivo para manter um buffer de capacidade é o tempo de espera do jogador. Quando slots de sessão de jogo estão prontos e em espera, leva segundos para obter novos jogadores em sessões de jogo. Se nenhum recurso estiver disponível, os jogadores devem esperar por sessões de jogo existentes serem encerradas ou novos recursos se tornarem disponíveis. Pode levar minutos para iniciar novas instâncias e processos de servidor.

Ao configurar o ajuste de escala automático baseado em destino, especifique o tamanho do buffer que você deseja que a frota mantenha. Como `PercentAvailableGameSessions` mede a porcentagem de recursos disponíveis, o tamanho real do buffer é uma porcentagem da capacidade total da frota. Amazon GameLift Servers adiciona ou remove instâncias para manter o tamanho do buffer de destino. Com um buffer grande, você minimiza o tempo de espera, mas também paga pelos recursos adicionais que talvez não use. Se os seus jogadores forem mais tolerantes com tempos de espera, você pode reduzir os custos definindo um buffer pequeno.

Para definir o ajuste de escala automático com base em destino

Console

1. Abra as [Amazon GameLift Servers console](#).

2. No painel de navegação, selecione Hospedagem, Frotas.
3. Na página Frotas, escolha o nome de uma frota ativa para abrir a página de detalhes da frota.
4. Escolha a guia Escalabilidade. Essa guia exibe as métricas de escalabilidade da frota e contém controles históricos para ajustar as configurações de escalabilidade atuais.
5. Em Capacidade de escalabilidade, verifique se os limites de tamanho mínimo e máximo são apropriados para a frota. Com o ajuste de escala automático ativado, a capacidade se ajusta entre esses dois limites.
6. Em Política de ajuste de escala automático baseada em destino, selecione Editar.
7. Na caixa de diálogo Editar política de ajuste de escala automático baseada em destino, em Porcentagem de sessões de jogo disponíveis, defina a porcentagem que você deseja manter e escolha Confirmar. Depois de confirmar as configurações, Amazon GameLift Servers adiciona uma nova política baseada em metas em Política de auto-scaling baseada em metas.

AWS CLI

1. Defina limites de capacidade. Defina os valores limite usando o [update-fleet-capacity](#) comando. Para obter mais informações, consulte [Defina limites Amazon GameLift Servers de capacidade](#).
2. Crie uma política. Abra uma janela de linha de comando e use o [put-scaling-policy](#) comando com as configurações de parâmetros da sua política. Para atualizar uma política existente, especifique o nome da política e forneça uma versão completa da política atualizada.

```
--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>
```

Exemplo: .

```
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"
```

Ajuste de escala automático com políticas com base nas regras

As políticas de escalabilidade baseadas em regras Amazon GameLift Servers fornecem controle refinado ao escalar automaticamente a capacidade de uma frota em resposta à atividade do jogador. Para cada política, é possível vincular a escalabilidade a uma das várias métricas da frota, identificar uma variação e personalizar a resposta de expansão ou evento de redução. As políticas com base em regras são úteis para completar a [escalabilidade com base em destino](#) para lidar com circunstâncias especiais.

Uma política baseada em regras afirma o seguinte: "Se a métrica de uma frota atingir ou ultrapassar um valor limite para um determinado período, altere a capacidade da frota em uma determinada quantidade." Este tópico descreve a sintaxe usada para estruturar uma declaração de política e fornece ajuda com a criação e o gerenciamento de políticas com base em regras.

Gerenciar políticas baseadas em regras

Crie, atualize ou exclua políticas baseadas em regras usando um AWS SDK ou o AWS Command Line Interface (AWS CLI) com a API de [serviço](#) para. Amazon GameLift Servers É possível visualizar todas as políticas ativas no console do Amazon GameLift Servers.

Para interromper temporariamente todas as políticas de escalabilidade de uma frota, use o AWS CLI comando [stop-fleet-actions](#).

Para criar ou atualizar uma política de escalabilidade baseada em regras (AWS CLI):

1. Defina limites de capacidade. Defina um ou ambos os valores limite usando o [update-fleet-capacity](#) comando. Para obter mais informações, consulte [Defina limites Amazon GameLift Servers de capacidade](#).
2. Crie uma política. Abra uma janela de linha de comando e use o [put-scaling-policy](#) comando com as configurações de parâmetros da sua política. Para atualizar uma política existente, especifique o nome da política e forneça uma versão completa da política atualizada.

```
--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>
```

Exemplo:

```
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
```

Para excluir uma regra de escalabilidade com base em política usando a AWS CLI:

- Abra uma janela de linha de comando e use o [delete-scaling-policy](#) comando com o ID da frota e o nome da política.

Exemplo:

```
aws gamelift delete-scaling-policy \
  --fleet-id fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa \
  --name "Scale up when AGS<50"
```

Regras de sintaxe para o ajuste de escala automático

Para construir uma declaração de política de escalabilidade baseada em regra, especifique seis variáveis:

Se *<metric name>* persistir *<comparison operator>* *<threshold value>**<evaluation period>*, altere a capacidade da frota usando *<adjustment type>* to/by *<adjustment value>*.

Por exemplo, esta declaração de política inicia um evento de expansão sempre que uma frota de capacidade extra é menor do que o que é necessário para lidar com 50 novas sessões de jogos:

Se `AvailableGameSessions` permanecer em `less than 50` para `10` minutes, altere a capacidade da frota usando `ChangeInCapacity` por `1` instances.

Nome da métrica

Para iniciar um evento de escalabilidade, vincule uma política de ajuste de escala automático para uma das seguintes métricas específicas da frota. Para obter descrições de métrica completa, consulte [Métricas do Amazon GameLift Servers para frotas](#).

- Ativar sessões de jogos
- Sessões de jogos ativas
- Sessões de jogos disponíveis
- Sessões de jogos disponíveis em porcentagem
- Instâncias ativas
- Sessões de jogadores disponíveis
- Sessões de jogadores atuais
- Instâncias ociosas
- Instâncias ociosas em porcentagem

Caso a frota esteja em uma fila de sessão de jogo, use as seguintes métricas:

- Profundidade da fila – O número de solicitações de sessão de jogo pendentes nesta frota é o melhor local de hospedagem disponível.
- Tempo de espera – Tempo de espera específico da frota. O período de tempo em que a solicitação de sessão de jogo pendente mais antigo está esperando para ser atendido. O tempo de espera da frota é igual ao tempo da solicitação atual mais antiga na fila.

Operador de comparação

Mostra Amazon GameLift Servers como comparar os dados métricos com o valor limite. Entre os operadores de comparação válidos estão maior que (`>`), menor que (`<`), maior que ou igual a (`>=`) e menor que ou igual a (`<=`).

Valor limite

Quando o valor de métrica especificado encontra ou cruza o valor de limite, ele pode iniciar um evento de escalabilidade. Esse valor é sempre um inteiro positivo.

Período de avaliação

A métrica deve atingir ou ultrapassar o valor limite em toda a duração do período de avaliação antes de iniciar um evento de escalabilidade. O tamanho do período de avaliação é consecutivo; se a métrica não atingir o limite, o período de avaliação recomeçará.

Tipo de ajuste e valor

Esse conjunto de variáveis trabalha em conjunto para Amazon GameLift Servers especificar como ajustar a capacidade da frota quando um evento de escalabilidade começa. Escolha um dos três tipos de ajustes possíveis:

- **Alteração na capacidade** – Aumente ou diminua a capacidade atual segundo o número especificado de instâncias. Defina o valor de ajuste para o número de instâncias a serem adicionadas a ou subtraídas da frota. Os valores positivos adicionam instâncias, enquanto os valores negativos removem instâncias. Por exemplo, um valor de "-10" reduz a frota por 10 instâncias, independentemente do tamanho total da frota.
- **Alteração percentual na capacidade** – Aumente ou diminua a capacidade atual segundo uma porcentagem especificada. Defina o valor de ajuste para a porcentagem na qual você deseja aumentar ou diminuir a capacidade da frota. Os valores positivos adicionam instâncias, enquanto os valores negativos removem instâncias. Por exemplo, para uma frota com 50 instâncias, uma alteração de porcentagem de "20" adiciona 10 instâncias à frota.
- **Capacidade exata** — Aumentar ou diminuir a capacidade atual para um valor específico. Defina o valor de ajuste para o número exato de instâncias que você deseja manter na frota.

Dicas para o ajuste de escala automático com base em regras

As sugestões a seguir podem ajudar a obter o máximo de ajuste de escala automático com políticas baseadas em regras.

Usar várias políticas

Você pode ter várias políticas de ajuste de escala automático para uma frota ao mesmo tempo. O cenário mais comum é ter uma política baseada em destino gerenciar a maioria dos requisitos de escalabilidade e usar políticas com base em regras para lidar com problemas em parâmetros. Não há limites no uso de várias políticas.

Com várias políticas, cada uma delas se comporta de forma independente. Não há como controlar a sequência de eventos de escalabilidade. Por exemplo, se você tiver várias políticas aumentando a escalabilidade, será possível que a atividade do jogador inicie vários eventos de escalabilidade

simultaneamente. Evite políticas que se iniciem mutuamente. Por exemplo, você poderá criar um loop infinito se aumentar e diminuir políticas que definam a capacidade além do limite uma da outra.

Definir as capacidades máxima e mínima

Cada frota tem um limite de capacidade máxima e mínima. Esse recurso é especialmente importante quando se usa o ajuste de escala automático. O ajuste de escala automático jamais define a capacidade como um valor fora desse intervalo. Por padrão, frotas recém-criadas têm um mínimo de 0 e um máximo de 1. Para a política de ajuste de escala automático afetar a capacidade conforme desejado, aumente o valor máximo.

A capacidade da frota também é limitada pelos limites do tipo de instância da frota e pelas cotas de serviço na sua. Conta da AWS Não é possível definir um mínimo e um máximo fora desses limites e cotas de conta

Acompanhar métricas depois de uma alteração na capacidade

Depois de alterar a capacidade em resposta a uma política de auto scaling, Amazon GameLift Servers espera 10 minutos antes de responder aos acionadores da mesma política. Essa espera dá Amazon GameLift Servers tempo para adicionar as novas instâncias, iniciar os servidores do jogo, conectar jogadores e começar a coletar dados das novas instâncias. Durante esse período, Amazon GameLift Servers avalia a política em relação à métrica e rastreia o período de avaliação da política, que é reiniciado após a ocorrência de um evento de escalabilidade. Isso significa que uma política de escalabilidade poderá iniciar outro evento de escalabilidade logo depois que o tempo de espera terminar.

Não há tempo de espera entre os eventos de escalabilidade que iniciam diferentes políticas de ajuste de escala automático.

Escale frotas de Amazon GameLift Servers contêineres

Uma das tarefas mais desafiadoras da hospedagem de jogos é escalar a capacidade para atender à demanda dos jogadores sem desperdiçar dinheiro com recursos que não sejam necessários. Em uma frota de contêineres gerenciada, você escala a capacidade da sua frota adicionando ou removendo instâncias da frota.

Quando você cria uma nova frota, Amazon GameLift Servers define a capacidade desejada da frota em uma instância e implanta uma instância na região de origem da frota. Para uma frota com vários locais, Amazon GameLift Servers implanta uma instância na região de origem e em cada

local remoto. Depois que o status da frota ACTIVE for atingido, você poderá aumentar a capacidade desejada para aumentar ou diminuir a capacidade desejada para reduzir a escala.

Você pode usar os recursos Amazon GameLift Servers de escalabilidade para alterar a capacidade manualmente ou configurar o escalonamento automático com base na demanda do jogador:

- Configure o escalonamento automático com rastreamento de alvos. Consulte [Ajuste de escala automático baseado em destino](#).
- Altere manualmente a capacidade da sua frota. Consulte [Defina manualmente a capacidade de uma Amazon GameLift Servers frota](#).

Ao escalar uma frota de contêineres, considere como a adição ou remoção de instâncias afeta a capacidade da frota de hospedar sessões de jogos e jogadores.

- Sessões de jogo por instância
 - Cada processo do servidor de jogos em execução em uma instância representa a capacidade de hospedar uma sessão de jogo.
 - Use essa fórmula para calcular o número de sessões de jogo que são executadas simultaneamente em uma instância de frota de contêineres:

```
[Game sessions per instance] = [# of game server processes per game server container] * [# of game server container groups per instance]
```

Se sua arquitetura de contêiner executa um processo de servidor de jogos simultaneamente no contêiner do servidor de jogos, as sessões de jogo por instância são iguais ao número de grupos de contêineres do servidor de jogos por instância.

- Para grupos de contêineres de servidores de jogos por instância, ligue [DescribeContainerFleet](#) para obter o `MaximumGameServerContainerGroupsPerInstance` valor `GameServerContainerGroupsPerInstance` ou.
- Jogadores por instância
 - Você decide o número de vagas de jogador a serem permitidas em cada sessão de jogo. Dependendo de como sua solução de hospedagem lida com o posicionamento da sessão de jogo, você pode definir jogadores por sessão de jogo em sua configuração de matchmaking ou em suas chamadas para iniciar a colocação de uma sessão de jogo.
 - Use essa fórmula para calcular o número de jogadores que podem jogar seu jogo simultaneamente em uma instância de frota de contêineres:

$$[\text{Players per instance}] = [\text{\# of game sessions per instance}] * [\text{\# of player slots per game session}]$$

Para obter a capacidade total atual de uma frota de contêineres, ligue para [DescribeFleetCapacity](#) ou [DescribeFleetLocation Capacity](#) para obter o número de grupos de contêineres de servidores de jogos na frota. Grupos ativos são aqueles que atualmente hospedam sessões de jogos. Grupos ociosos estão prontos para sediar uma nova sessão de jogo. Multiplique esses valores pelo número de processos do servidor por grupo de contêineres do servidor do jogo.

Preparando seu jogo para o lançamento com Amazon GameLift Servers hospedagem

Use as listas de verificação a seguir para validar cada fase de implantação do jogo. Os itens marcados como [Críticos] são essenciais para o lançamento da produção.

Baixe e preencha o questionário de Amazon GameLift Servers lançamento, que está disponível no [Amazon GameLift Serversconsole](#). Queremos que todos os desenvolvedores de jogos Amazon GameLift Servers tenham um dia de lançamento tranquilo, e as informações solicitadas nos ajudam a se preparar para o próximo teste de carga, lançamento temporário ou lançamento público. Planeje enviar o questionário preenchido pelo menos três meses antes de realizar o primeiro teste de carga.

Tópicos

- [Preparar o jogo](#)
- [Preparar para testes](#)
- [Preparar para o início](#)
- [Planejar atualizações pós-lançamento](#)

Preparar o jogo

- [Crítico] Verifique se você concluiu todas as [etapas do roteiro de desenvolvimento](#) da solução de hospedagem e se tem todos os componentes necessários em funcionamento, incluindo um servidor de jogos integrado, um serviço de backend para os clientes de jogo, frotas de hospedagem e um método de posicionamento de sessões de jogo (como uma fila).
- [Crítico] [Crie funções AWS Identity and Access Management \(IAM\)](#) que permitam que seu servidor de jogos acesse outros AWS recursos durante a execução.
- [Crítico] Projete e implemente o failover para outros recursos de hospedagem, conforme necessário.
- [Planeje a distribuição de frotas para os locais de destino](#), considerando a fila e a estrutura da frota do seu jogo.
- [Automatize sua implantação](#) usando infraestrutura como código (IaC) com AWS CloudFormation e AWS Cloud Development Kit (AWS CDK)
- [Colete registros e análises](#) usando a Amazon CloudWatch e o Amazon Simple Storage Service (Amazon S3).

Preparar para testes

- [Crítico] [Solicite aumentos Amazon GameLift Servers de cotas de serviço](#) e outras AWS service (Serviço da AWS) cotas para que seu ambiente ativo possa se expandir de acordo com as necessidades de produção.
- [Crítico] Verifique se as portas abertas nas frotas ativas correspondem ao intervalo de portas que os servidores poderiam usar.
- [Crítico] Feche a porta RDP 3389 e a porta SSH 22.
- Desenvolva um plano para o DevOps gerenciamento do seu jogo. Se você estiver usando o Amazon CloudWatch Logs ou métricas CloudWatch personalizadas da Amazon, defina alarmes para problemas graves ou críticos na frota de servidores. Simule falhas e teste os runbooks.
- Verifique se os recursos de computação que você está usando são compatíveis com o número de processos do servidor que você deseja executar simultaneamente em cada computação.
- [Ajuste sua política de escalabilidade](#) para ser mais conservadora no início e fornecer mais capacidade ociosa do que considera ser necessário. É possível otimizar o custo posteriormente. Considere o uso de uma política de escalabilidade com base em destino com 20% de capacidade ociosa.
- PoisFlexMatch, use [as regras de latência](#) para combinar jogadores que estão geograficamente próximos uns dos outros. Teste como isso se comporta sob carga com dados de latência sintéticos do seu cliente de teste de carga.
- Teste a carga da autenticação do jogador e da infraestrutura da sessão de jogo para observar se ela é dimensionada de forma eficaz para atender à demanda.
- Verifique se um servidor deixado em execução por vários dias ainda pode aceitar conexões.
- Eleve o nível do seu Suporte plano para Business ou Enterprise para que ele AWS possa responder a você durante problemas ou interrupções.

Preparar para o início

- [Crítico] [Defina a política de proteção da frota](#) para proteção total em todas as frotas ativas, para que a redução não interrompa as sessões ativas do jogo.
- [Crítico] [Defina tamanhos máximos de frota](#) altos o suficiente para acomodar o pico de demanda previsto, no mínimo. É recomendável dobrar seu tamanho máximo para uma demanda imprevista.
- Incentive toda a equipe de desenvolvimento a participar do evento de lançamento e monitorar o lançamento do jogo em uma sala de lançamento.

- Monitore a latência e a experiência do jogador.

Planejar atualizações pós-lançamento

- [Ajuste a política de ajuste de escala](#) para minimizar a capacidade ociosa com base no uso dos jogadores.
- [Modifique FlexMatch as regras](#) ou [adicione locais de hospedagem](#) com base nos dados de latência do jogador e nos requisitos revisados.
- Otimize a configuração do tempo de execução para executar o maior número possível de sessões de jogos em cada recurso de computação. Maximizar a eficiência do desempenho dessa forma pode afetar diretamente os custos da frota, pois você poderá executar mais processos de servidor com os mesmos recursos de computação.
- [Use seus dados de análise](#) para impulsionar o desenvolvimento contínuo, melhorar a experiência do jogador e a longevidade do jogo e otimizar a monetização.

Gerenciamento Amazon GameLift Servers recursos de hospedagem usando AWS CloudFormation

Você pode usar AWS CloudFormation para gerenciar seu Amazon GameLift Servers recursos. Em AWS CloudFormation, você cria um modelo que modela cada recurso e, em seguida, usa o modelo para criar seus recursos. Para atualizar os recursos, você faz as alterações em seu modelo e usa AWS CloudFormation para implementar as atualizações. É possível organizar os recursos em grupos lógicos, chamados pilhas e conjuntos de pilhas.

Usando AWS CloudFormation para manter seu Amazon GameLift Servers os recursos de hospedagem oferecem uma maneira mais eficiente de gerenciar conjuntos de AWS recursos. É possível usar o controle de versão para monitorar alterações de modelo ao longo do tempo e atualizações de coordenadas feitas por vários membros de equipe. Também é possível reutilizar modelos. Por exemplo, ao implantar um jogo em várias regiões, é possível usar o mesmo modelo para criar recursos idênticos em cada região. Também é possível usar esses modelos para implantar o mesmo conjunto de recursos em outra partição.

Para obter mais informações sobre AWS CloudFormation, consulte o [Guia AWS CloudFormation do usuário](#). Para visualizar as informações do modelo para Amazon GameLift Servers recursos, veja o [Amazon GameLift Servers referência do tipo de recurso](#).

Práticas recomendadas

Para obter orientações detalhadas sobre o uso AWS CloudFormation, consulte as [AWS CloudFormation melhores práticas](#) no Guia AWS CloudFormation do usuário. Além disso, essas melhores práticas têm especial relevância com Amazon GameLift Servers.

- Gerencie os recursos de forma consistente usando o AWS CloudFormation. Se você alterar seus recursos fora dos AWS CloudFormation seus recursos, eles ficarão fora de sincronia com seus modelos de recursos.
- Use AWS CloudFormation pilhas e conjuntos de pilhas para gerenciar com eficiência vários recursos.
 - Use pilhas para gerenciar grupos de recursos conectados. Por exemplo, uma pilha que contém uma compilação, uma frota que faça referência à compilação e um alias que faça referência à frota. Se você atualizar seu modelo para substituir uma compilação, AWS CloudFormation

substituirá as frotas conectadas à compilação. AWS CloudFormation em seguida, atualiza os aliases existentes para apontar para as novas frotas. Para obter mais informações, consulte [Como trabalhar com pilhas](#) no Guia do usuário do AWS CloudFormation .

- Use conjuntos de AWS CloudFormation pilhas se estiver implantando pilhas idênticas em várias regiões ou contas. Para mais informações, consulte [Como trabalhar com conjuntos de pilhas](#) no Guia do usuário do AWS CloudFormation .
- Se estiver usando instâncias spot, inclua uma frota sob demanda como backup. Recomendamos configurar os modelos com duas frotas em cada região, uma frota com instâncias spot e uma frota com instâncias sob demanda.
- Agrupe os recursos específicos do local e os recursos globais em pilhas separadas ao gerenciar recursos em vários locais.
- Posicione os recursos globais próximos aos serviços que os utilizam. Recursos como filas e configurações de criação de jogos tendem a receber um alto volume de solicitações de origens específicas. Ao posicionar os recursos próximos à origem dessas solicitações, você minimiza o tempo de viagem da solicitação e pode melhorar o desempenho geral.
- Coloque a configuração de criação de jogos na mesma região da fila de sessões de jogo usada.
- Crie um alias separado para cada frota na pilha.

Usando AWS CloudFormation pilhas

Recomendamos que as seguintes estruturas sejam usadas ao configurar AWS CloudFormation pilhas para Amazon GameLift Servers recursos. A estrutura de pilha ideal varia dependendo se você está implantando o jogo em um local ou em vários locais.

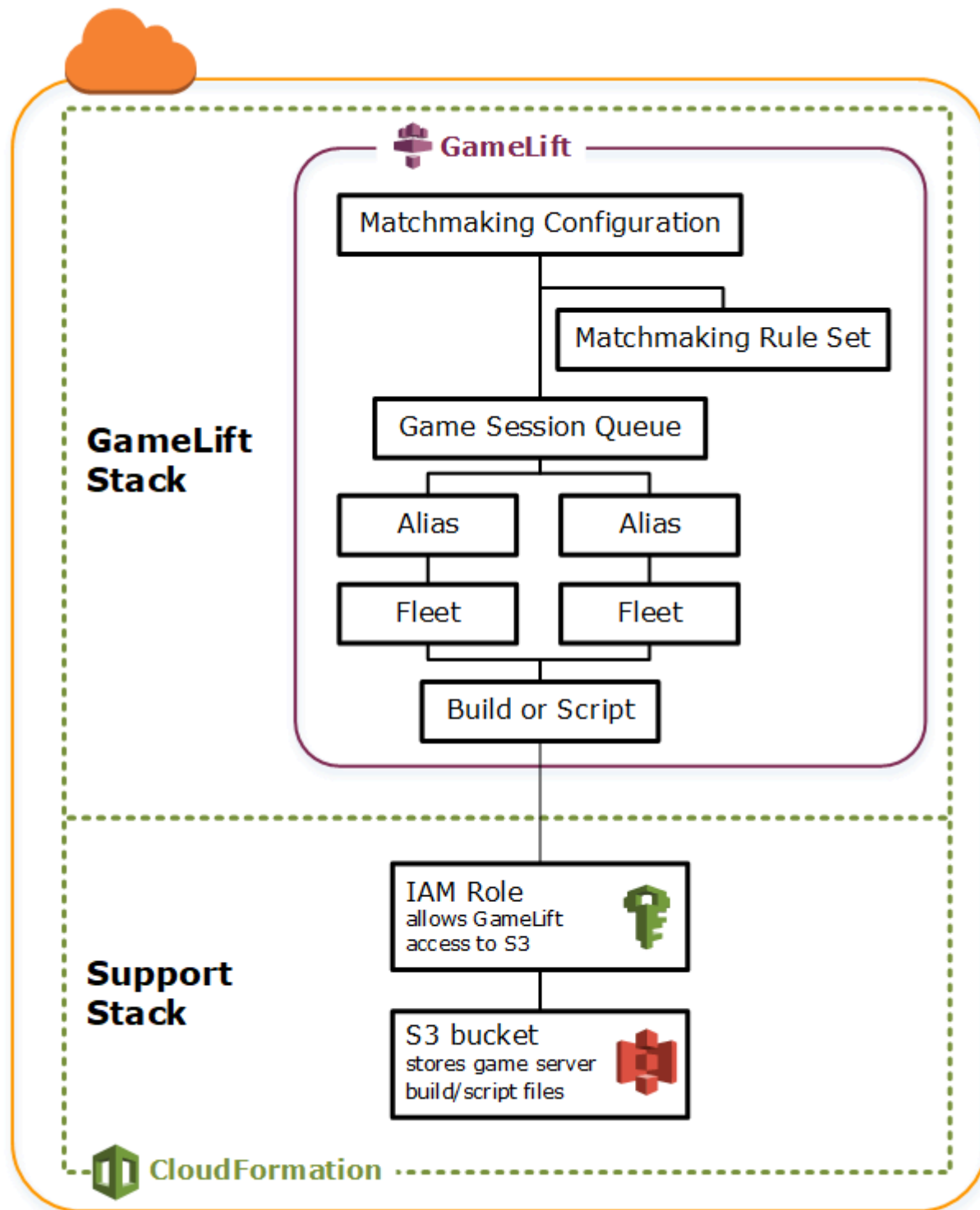
Pilhas para um único local

Para gerenciar Amazon GameLift Servers recursos em um único local, recomendamos uma estrutura de duas pilhas:

- Support stack — Essa pilha contém recursos que seu Amazon GameLift Servers os recursos dependem de. No mínimo, essa pilha deve incluir o bucket do S3 onde você armazena o servidor de jogo personalizado ou arquivos de script Realtime. A pilha também deve incluir uma função do IAM que forneça Amazon GameLift Servers permissão para recuperar seus arquivos do bucket do S3 ao criar um Amazon GameLift Servers recurso de construção ou script. Essa pilha também pode conter outros AWS recursos usados com seu jogo, como tabelas do DynamoDB, clusters do Amazon Redshift e funções Lambda.

- **Amazon GameLift Servers pilha** — Esta pilha contém todas as suas Amazon GameLift Servers recursos, incluindo a construção ou o script, um conjunto de frotas, aliases e fila de sessões de jogo. AWS CloudFormation cria um recurso de compilação ou script com arquivos armazenados no local do bucket do S3 e implanta a compilação ou o script em um ou mais recursos da frota. Cada frota deve ter um alias correspondente. A fila de sessões de jogo faz referência a alguns ou a todos os aliases da frota. Se você estiver usando FlexMatch para matchmaking, esta pilha também contém uma configuração de matchmaking e um conjunto de regras.

O diagrama abaixo ilustra uma estrutura de duas pilhas para implantar recursos em uma única região. AWS



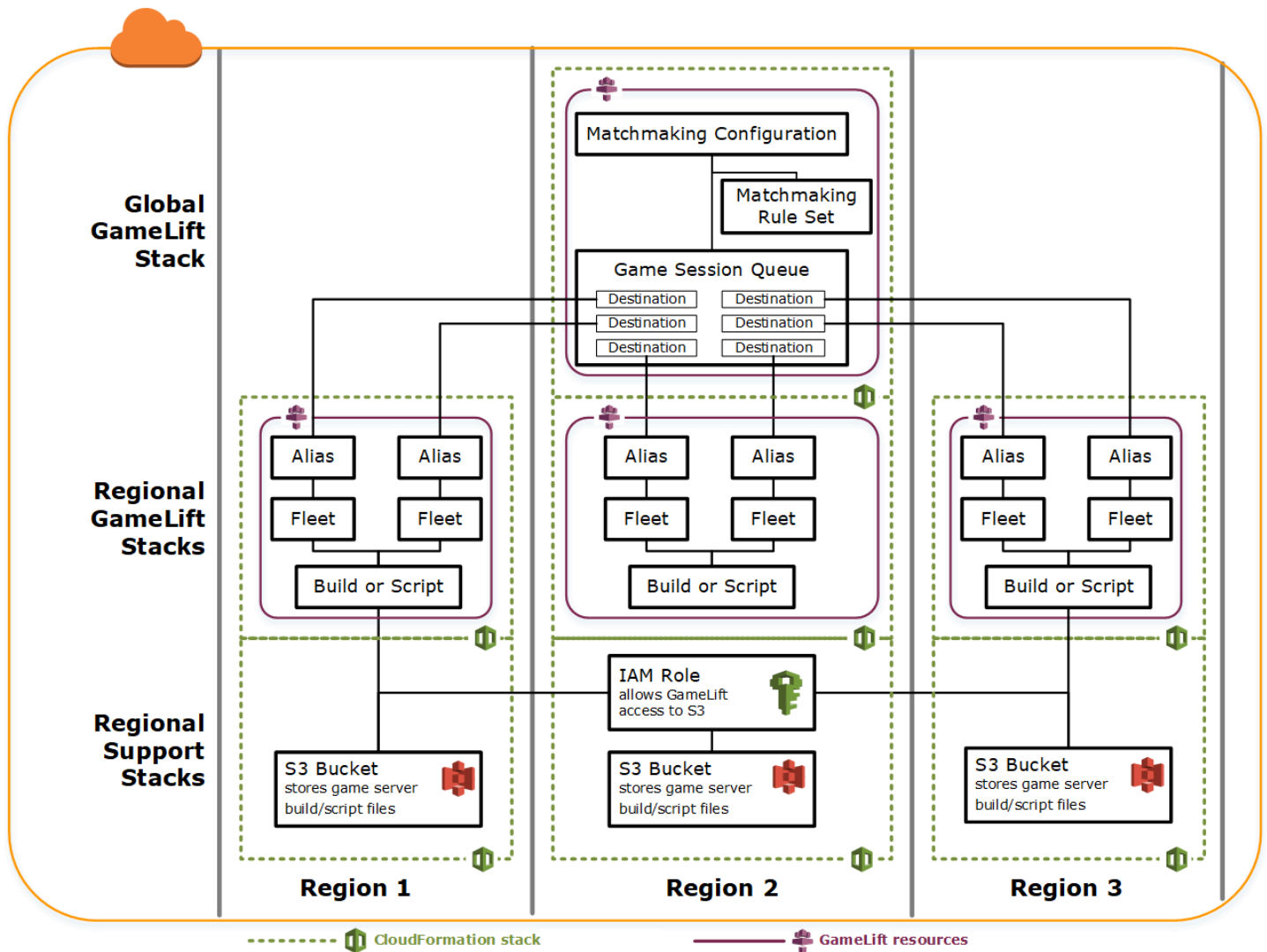
Pilhas para várias regiões

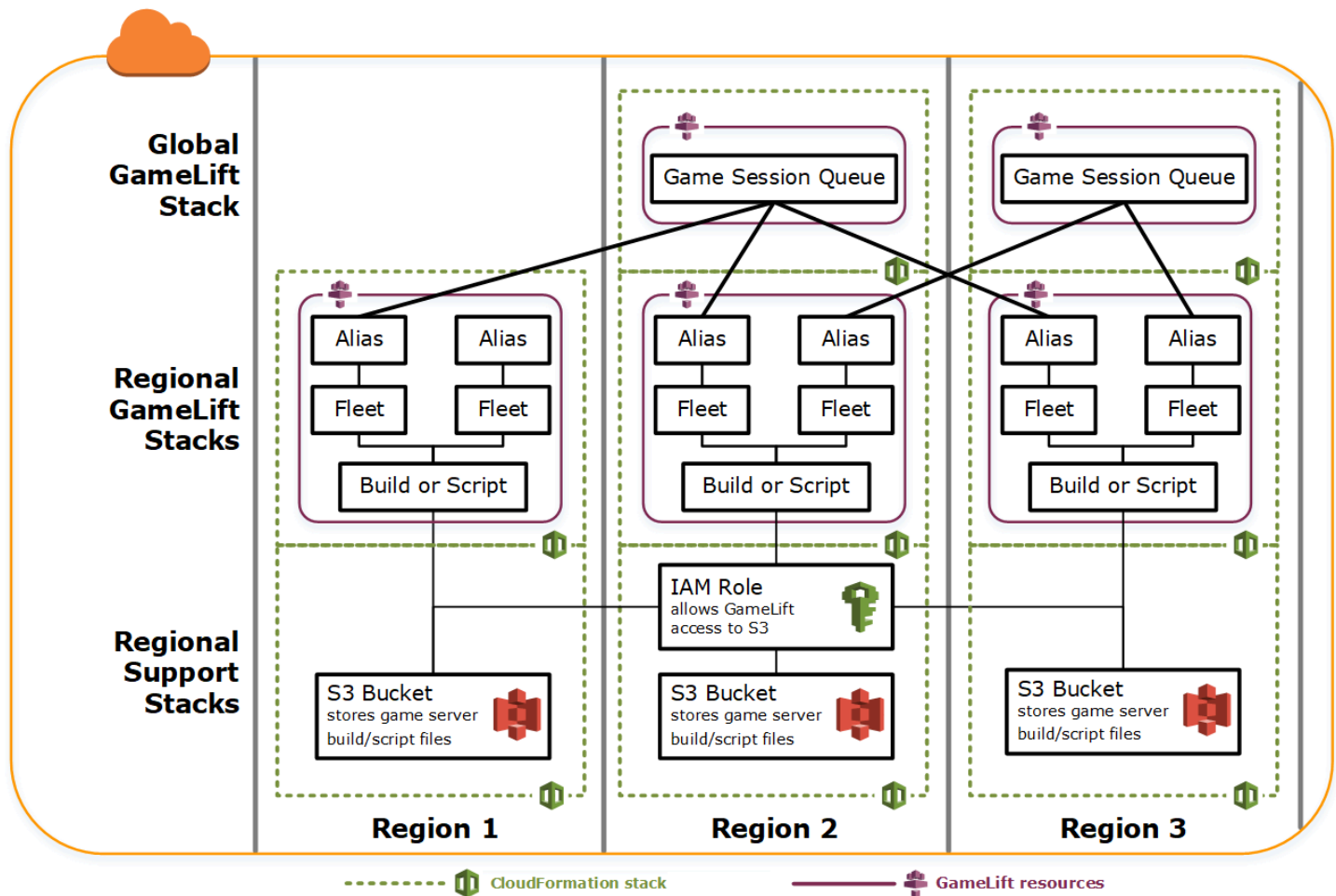
Ao implantar o jogo em mais de uma região, lembre-se de como os recursos podem interagir entre regiões. Alguns recursos, como Amazon GameLift Servers frotas, só podem referenciar outros recursos na mesma região. Outros recursos, como um Amazon GameLift Servers fila, são

independentes de região. Para gerenciar Amazon GameLift Servers recursos em várias regiões, recomendamos a estrutura a seguir.

- **Pilhas de suporte regional** — Essas pilhas contêm recursos que seu Amazon GameLift Servers os recursos dependem de. Essa pilha deve incluir o bucket do S3 onde você armazena o servidor de jogo personalizado ou os arquivos de script em tempo real. Ele também pode conter outros AWS recursos para o seu jogo, como tabelas do DynamoDB, clusters do Amazon Redshift e funções do Lambda. Muitos desses recursos são específicos da região e, sendo assim, você deve criá-los em todas as regiões. Amazon GameLift Servers também precisa de uma função do IAM que permita o acesso a esses recursos de suporte. Como um perfil do IAM é agnóstico em relação à região, só é necessário um recurso de função colocado em qualquer região e referenciado em todas as outras pilhas compatíveis.
- **Regional Amazon GameLift Servers pilhas** — Essa pilha contém o Amazon GameLift Servers recursos que devem existir em cada região em que seu jogo está sendo implantado, incluindo a construção ou o script, um conjunto de frotas e aliases. AWS CloudFormation cria um recurso de compilação ou script com arquivos em um local de bucket do S3 e implanta a construção ou o script em um ou mais recursos da frota. Cada frota deve ter um alias correspondente. A fila de sessões de jogo faz referência a alguns ou a todos os aliases da frota. É possível manter um modelo para descrever esse tipo de pilha e usá-lo para criar conjuntos idênticos de recursos em cada região.
- **Global Amazon GameLift Servers pilha** — Esta pilha contém sua fila de sessões de jogo e recursos de matchmaking. Esses recursos podem estar localizados em qualquer região e geralmente são colocados na mesma região. A fila pode fazer referência a frotas ou a aliases que estão localizados em qualquer região. Para colocar filas adicionais em regiões diferentes, crie pilhas globais adicionais.

Os diagramas abaixo ilustram uma estrutura de várias pilhas para implantar recursos em várias regiões. AWS O primeiro diagrama mostra uma estrutura de uma única fila de sessão de jogo. O segundo diagrama mostra uma estrutura com várias filas.





Atualizar compilações

Amazon GameLift Servers as construções são imutáveis, assim como a relação entre uma construção e uma frota. Como resultado, ao atualizar os recursos de hospedagem para usar um novo conjunto de arquivos de compilação de jogos, é necessário fazer o seguinte:

- Crie uma compilação usando o novo conjunto de arquivos (substituição).
- Crie um conjunto de frotas para implantar a nova compilação do jogo (substituição).
- Redirecione os aliases para apontar para as novas frotas (atualização sem interrupção).

Para obter mais informações, consulte [Atualizar comportamentos de recursos de pilha](#) no Guia do usuário do AWS CloudFormation .

Implantar atualizações de compilação automaticamente

Ao atualizar uma pilha contendo recursos relacionados de construção, frota e alias, o AWS CloudFormation comportamento padrão é executar automaticamente essas etapas em sequência. Acione essa atualização primeiro fazendo upload dos novos arquivos de compilação para um novo local do S3. Em seguida, você modifica seu modelo de AWS CloudFormation compilação para apontar para o novo local do S3. Ao atualizar a pilha com o novo local do S3, isso aciona a seguinte sequência do AWS CloudFormation :

1. Recupera os novos arquivos do S3, valida os arquivos e cria um novo Amazon GameLift Servers construir.
2. Atualiza a referência de compilação no modelo de frota, que aciona a criação da frota.
3. Depois que as novas frotas estiverem ativas, atualiza a referência da frota no alias, o que aciona o alias para atualizar a fim de atingir as novas frotas.
4. Apaga a frota antiga.
5. Exclui a compilação antiga.

Se a fila de sessões de jogo usar aliases de frota, o tráfego de jogadores será automaticamente alterado para as novas frotas assim que os aliases forem atualizados. As frotas antigas são gradualmente drenadas de jogadores à medida que as sessões de jogo terminam. A escalabilidade automática lida com a tarefa de adicionar e remover instâncias de cada conjunto de frotas à medida que o tráfego de jogadores flutua. Como alternativa, é possível especificar uma contagem inicial de instâncias desejada a fim de acelerar rapidamente para a troca e habilitar a escalabilidade automática mais tarde.

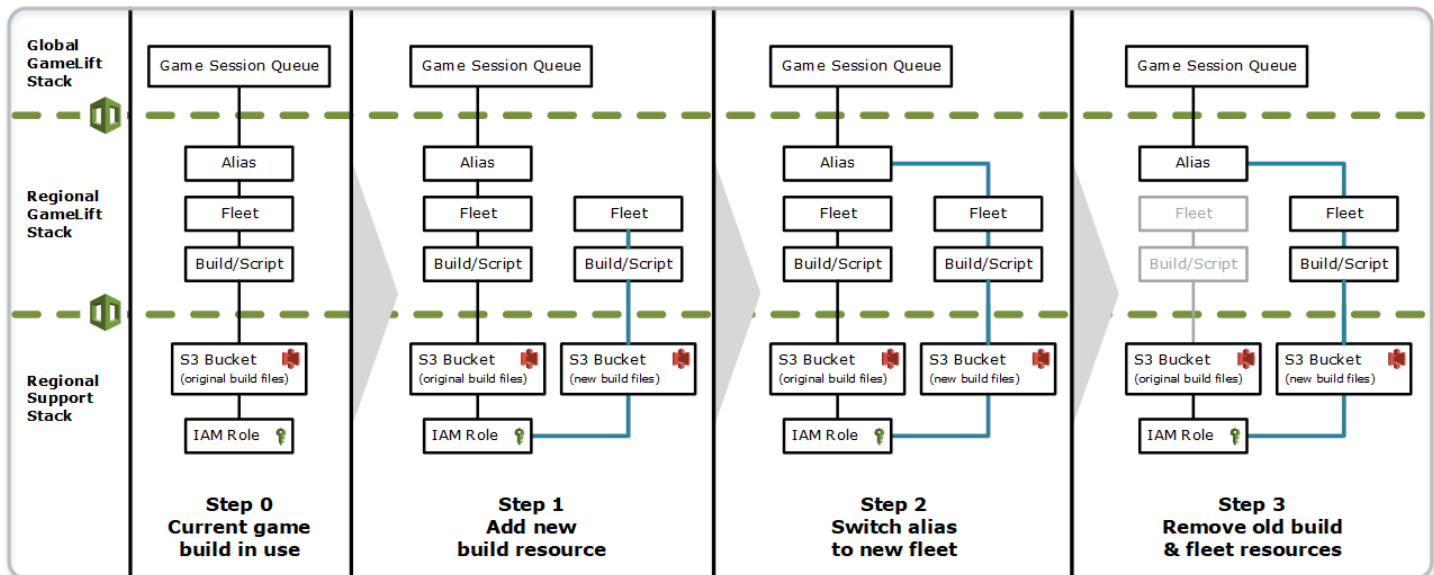
Você também pode AWS CloudFormation reter recursos em vez de excluí-los. Para obter mais informações, consulte [RetainResources](#) na Referência de APIs do AWS CloudFormation .

Implantar atualizações de compilação manualmente

Se quiser ter mais controle sobre quando novas frotas são ativadas para os jogadores, há algumas opções. Você pode optar por gerenciar aliases manualmente usando o Amazon GameLift Servers console ou a CLI. Como alternativa, em vez de atualizar o modelo de compilação para substituir as compilações e as frotas, é possível adicionar um segundo conjunto de definições de compilação e de frota ao modelo. Quando você atualiza o modelo, AWS CloudFormation cria um segundo recurso de compilação e as frotas correspondentes. Como os recursos existentes não são substituídos, eles não são excluídos e os aliases permanecem apontando para as frotas originais.

A principal vantagem desta abordagem é que ela oferece flexibilidade. É possível criar recursos separados para a nova versão da compilação, testar os novos recursos e controlar quando a nova frota é acionada para os jogadores. Uma possível desvantagem é que ela requer duas vezes mais recursos em cada região por um breve período.

O diagrama a seguir ilustra esse processo.



Como funcionam as reversões

Ao executar uma atualização de recurso, se qualquer etapa não for concluída com êxito, o AWS CloudFormation iniciará automaticamente uma reversão. Esse processo reverte cada etapa em sequência, excluindo os recursos recém-criados.

Se precisar acionar manualmente uma reversão, altere a chave de localização do S3 do modelo de compilação de volta para o local original e atualize a pilha. Um novo Amazon GameLift Servers a construção e a frota são criadas, e o alias muda para a nova frota depois que a frota estiver ativa. Se estiver gerenciando aliases separadamente, será necessário mudá-los a fim de apontar para as novas frotas.

Para obter mais informações sobre como lidar com uma reversão que falha ou trava, consulte [Continuar revertendo uma atualização](#) no Guia do usuário do AWS CloudFormation .

Monitoramento Amazon GameLift Servers

Se você estiver usando Amazon GameLift Servers FleetIQ como um recurso independente da Amazon EC2, consulte [Segurança na Amazon EC2 no](#) Guia do EC2 usuário da Amazon.

O monitoramento é uma parte importante da manutenção da confiabilidade, disponibilidade e desempenho do Amazon GameLift Servers e suas outras AWS soluções. Há três usos principais para métricas com Amazon GameLift Servers: para monitorar a integridade do sistema e configurar alarmes, monitorar o desempenho e o uso do servidor de jogos e gerenciar a capacidade usando o escalonamento manual ou automático.

AWS fornece as seguintes ferramentas de monitoramento para observar Amazon GameLift Servers, relate quando algo está errado e execute ações automáticas quando apropriado:

- Amazon GameLift Servers console — Use a interface gráfica para gerenciar seu Amazon GameLift Servers recursos e acompanhe a atividade de hospedagem de jogos.
- Amazon CloudWatch — Você pode monitorar Amazon GameLift Servers métricas em tempo real, bem como métricas para outros AWS recursos e aplicativos que você está executando nos AWS serviços. CloudWatch oferece um conjunto de recursos de monitoramento, incluindo ferramentas para criar painéis personalizados e a capacidade de definir alarmes que notificam ou agem quando uma métrica atinge um limite especificado.
- AWS CloudTrail— captura todas as chamadas de API e eventos relacionados feitos por ou em nome de sua AWS conta para Amazon GameLift Servers e outros AWS serviços. Os dados são fornecidos como arquivos de log para um bucket do Amazon S3 especificado por você. Você pode identificar quais usuários e contas ligaram AWS, o endereço IP de origem a partir do qual as chamadas foram feitas e quando elas ocorreram.
- Registros de sessão de jogo – Você poderá enviar mensagens personalizadas do servidor para suas sessões de jogo em arquivos de log armazenados no Amazon S3.

Tópicos

- [Acompanhe a hospedagem de jogos no Amazon GameLift Servers console](#)
- [Monitore Amazon GameLift Servers com a Amazon CloudWatch](#)
- [Registro em log Amazon GameLift Servers Chamadas de API com AWS CloudTrail](#)
- [Registrando mensagens do servidor em Amazon GameLift Servers](#)

Acompanhe a hospedagem de jogos no Amazon GameLift Servers console

Use o comando Amazon GameLift Servers console para visualizar e gerenciar seus recursos de hospedagem de jogos e atividades contínuas de hospedagem quase em tempo real. O console oferece uma interface gráfica para a maioria das funcionalidades da API de serviço para Amazon GameLift Servers. Você pode usar o console para:

- Use o painel para obter um instantâneo de alto nível. Você pode ver os números e o status atual de todos os seus Amazon GameLift Servers recursos de hospedagem e siga os links para obter detalhes sobre recursos individuais.
- Gerencie recursos individuais de hospedagem. Você pode criar, visualizar e excluir todos Amazon GameLift Servers recursos e atualize suas propriedades mutáveis. Você também pode ver certos tipos de atividades de hospedagem, como eventos e métricas de desempenho.
- Interaja com o jogo e a atividade da sessão do jogador. Você pode monitorar a sessão do jogo e a atividade da sessão do jogador por frota e usar essas informações para solucionar problemas na sessão do jogo. Veja os detalhes de uma sessão de jogo, veja as sessões dos jogadores para cada sessão de jogo e pesquise a atividade do jogador em várias sessões de jogo. Você também pode encerrar sessões de jogo individuais conforme necessário.

Tópicos

- [Painel de hospedagem no Amazon GameLift Servers painel](#)
- [O servidor de jogos é construído no Amazon GameLift Servers painel](#)
- [Amazon GameLift ServersRealtimescripts no Amazon GameLift Servers console](#)
- [Frotas no Amazon GameLift Servers console](#)
- [Detalhes da frota no Amazon GameLift Servers console](#)
- [Sessões de jogos e jogadores no Amazon GameLift Servers console](#)
- [Aliases no Amazon GameLift Servers console](#)
- [Filas de sessões de jogo no Amazon GameLift Servers console](#)

Painel de hospedagem no Amazon GameLift Servers painel

Use o comando Amazon GameLift Servers painel do console para obter um instantâneo de alto nível sobre o status atual do Amazon GameLift Servers recursos de hospedagem em sua AWS conta. A Amazon GameLift Servers o painel fornece uma visão do seguinte:

- O número de compilações nos status Pronto, Inicializado e Falha. Escolha Exibir compilações para obter detalhes sobre as compilações na sua região atual.
- O número de frotas em todos os status. Escolha Exibir frotas para obter detalhes sobre as frotas em sua região atual.
- Seus recursos atuais.
- Anúncios de novos recursos e serviços.

Para abrir o Amazon GameLift Servers painel

- No [Amazon GameLift Servers console](#), no painel de navegação, escolha Painel.

No painel, você poderá:

- Preparar seu jogo para o lançamento escolhendo Preparar para o lançamento e preencher o questionário de lançamento correspondente.
- Solicitar aumentos de cota de serviço em preparação para lançamentos ou em resposta a lançamentos escolhendo Exibir cotas de serviço.
- Visualizar postagens no blog e informações detalhadas sobre novos recursos escolhendo o link no Destaque de recursos.



GameLift > Dashboard

Dashboard

Build status overview

Viewing data for all builds in N. Virginia region.

 Ready
1

 Initialized
0

 Failed
0

[View builds](#)

Fleet status overview

Viewing data for all fleets in N. Virginia region.

 Active
0

 Deleting
0

 In progress
0

 New
0

 Error
0

 Terminated
0

[View fleets](#)

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

O servidor de jogos é construído no Amazon GameLift Servers painel

Na página Construções do [Amazon GameLift Servers console](#), você pode visualizar informações e gerenciar todas as compilações de servidores de jogos que você enviou para Amazon GameLift Servers para implantação em EC2 frotas gerenciadas. No painel de navegação, escolha Hosting, Managed EC2, Builds.

A página Compilações mostra as informações a seguir para cada compilação. Você pode ajustar o conteúdo da tabela conforme necessário usando a ferramenta Preferências (veja o ícone



no canto superior direito da tabela). As preferências personalizadas são salvas no usuário da sua conta AWS e aplicadas automaticamente sempre que você visualizar esta página.

 Note

A página Construções mostra somente as construções na sua AWS região atual.

- Nome – O nome associado à compilação cujo upload foi feito.
- Status – O status da compilação. Exibe uma das três mensagens de status:
 - Inicializado – O upload não foi iniciado ou ainda está em andamento.
 - Pronto – A compilação está pronta para a criação da frota.
 - Falha — A compilação atingiu o tempo limite antes Amazon GameLift Servers recebeu os binários.
- Hora de criação — A data e a hora em que você fez o upload da compilação Amazon GameLift Servers.
- ID da compilação – ID exclusivo atribuído à compilação no upload.
- Versão – O rótulo da versão associado à compilação cujo upload foi feito.
- Sistema operacional – O sistema operacional no qual a compilação é executada. O sistema operacional de compilação determina qual sistema operacional Amazon GameLift Servers instala nas instâncias de uma frota.
- Tamanho — O tamanho, em megabytes (MB), do arquivo de compilação enviado para Amazon GameLift Servers.
- Frotas – O número de frotas implantadas com a compilação.

Nesta página, você poderá fazer o seguinte:

- Visualize detalhes da compilação. Escolha o nome de uma compilação para abrir a página de detalhes da compilação.
- Crie uma nova frota com base em uma compilação. Selecione uma compilação e, em seguida, escolha Criar frota.
- Filtre e classifique a lista de compilações. Use os controles na parte superior da tabela.
- Exclua uma compilação. Selecione uma compilação e escolha Excluir.

Detalhes da compilação

Na página Compilações, escolha o nome de uma compilação para abrir sua página de detalhes. A seção Visão geral da página de detalhes exibe as mesmas informações de resumo da compilação que a página Compilações. A seção Frotas mostra uma lista das frotas criadas com a compilação, incluindo as mesmas informações resumidas da [página Frotas](#).

Amazon GameLift ServersRealtimescripts no Amazon GameLift Servers console

Na página Scripts do [Amazon GameLift Serversconsole](#), você pode visualizar informações e gerenciar todos os Amazon GameLift Servers Realtime scripts que você enviou Amazon GameLift Servers para implantação em EC2 frotas gerenciadas. No painel de navegação, escolha Hospedagem, Scripts.

A página Scripts mostra as seguintes informações para cada script. Você pode ajustar o conteúdo da tabela conforme necessário usando a ferramenta Preferências (veja o ícone



no canto superior direito da tabela). As preferências personalizadas são salvas no usuário da sua conta AWS e aplicadas automaticamente sempre que você visualizar esta página.

Note

A página Scripts mostra scripts somente na sua AWS região atual.

- Nome – O nome associado ao script carregado.
- ID – O ID exclusivo atribuído ao script no upload.
- Versão – O rótulo da versão associado ao script cujo upload foi feito.
- Tamanho — O tamanho, em megabytes (MB), do arquivo de script enviado para. Amazon GameLift Servers
- Hora de criação — A data e a hora em que você fez o upload do scriptAmazon GameLift Servers.
- Frotas – O número de frotas implantadas com o script.

Nesta página, você poderá fazer o seguinte:

- Visualize detalhes do script. Escolha o nome de uma compilação para abrir a página de detalhes do script.
- Crie uma nova frota com base em um script. Selecione um script e escolha Criar frota.
- Filtre e classifique a lista de script. Use os controles na parte superior da tabela.
- Exclua um script. Selecione um script e escolha Excluir.

Detalhes do script

Na página Scripts, escolha o nome de um script para abrir sua página de detalhes. A seção Visão geral da página de detalhes exibe as mesmas informações resumidas do script da página Compilações. A seção Frotas mostra uma lista das frotas criadas com o script, incluindo as mesmas informações resumidas da [página Frotas](#).

Frotas no Amazon GameLift Servers console

Você pode ver informações sobre todas as frotas criadas para hospedar seus jogos em Amazon GameLift Servers em sua AWS conta. A lista mostra as frotas criadas na sua região atual. Na página Fleets (Frotas), crie uma nova frota ou visualize detalhes adicionais sobre uma frota. A [página de detalhes](#) da frota contém informações de uso, métricas, dados da sessão do jogo e dados da sessão do jogador. Também é possível editar um registro de frota ou excluir uma frota.

Para visualizar a página Frotas, escolha Frotas no painel de navegação.

A página Frotas exibe, por padrão, as informações de resumo a seguir. Você pode ajustar o conteúdo da tabela conforme necessário usando a ferramenta Preferências (veja o ícone



no canto superior direito da tabela). As preferências personalizadas são salvas no usuário da sua conta AWS e aplicadas automaticamente sempre que você visualizar esta página.

- Nome – Nome amigável dado para a frota.
- Status – O status atual da frota, que pode ser um destes estados: Nova, Download em andamento, Compilação em andamento e Ativa.
- CHora de criação – A data e hora em que a frota foi criada.
- Tipo de computação – O tipo do computador usado para hospedar os jogos. Uma frota pode ser uma EC2 frota gerenciada ou uma frota em qualquer lugar.

- Tipo de instância — O tipo de EC2 instância da Amazon, que determina a capacidade computacional das instâncias da frota.
- Instâncias ativas — O número de EC2 instâncias em uso para a frota.
- Instâncias desejadas — O número de EC2 instâncias a serem mantidas ativas.
- Sessões de jogos – O número de sessões de jogos ativas em execução na frota. Os dados têm um atraso de cinco minutos.

Detalhes da frota no Amazon GameLift Servers console

Acesse a página de detalhes Frota pelo painel ou a página Frotas selecionando no nome da frota.

Na página de detalhes da frota, realize as seguintes ações:

- Atualize os atributos, as configurações de porta e a configuração de runtime de uma frota.
- Adicionar ou remover localizações da frota.
- Altere as configurações de capacidade da frota.
- Defina ou altere o ajuste de escala automático do rastreamento de destino.
- Exclua uma frota.

Detalhes

Configurações da frota

- ID da frota – Identificador exclusivo atribuído à frota.
- Nome – O nome da frota.
- ARN – O identificador atribuído à frota. O ARN de uma frota a identifica como Amazon GameLift Servers recurso e especifica a região e a AWS conta.
- Descrição – Uma breve descrição identificável da frota.
- Status – Status atual da frota, que pode ser Nova, Download em andamento, Compilação em andamento e Ativa.
- Hora da criação – A data e a hora em que a frota foi criada.
- Hora de término – A data e a hora em que a frota foi encerrada. Isso ficará em branco se a frota ainda estiver ativa.
- Tipo de frota – Indica se a frota usa instâncias sob demanda ou spot.

- EC2 tipo — Tipo [de EC2 instância](#) da Amazon selecionado para a frota quando ela foi criada.
- Função da instância — uma função AWS do IAM que gerencia o acesso aos seus outros AWS recursos, caso um tenha sido fornecido durante a criação da frota.
- Certificado TLS – Se a frota está habilitada ou desativada para usar um certificado TLS para autenticar um servidor de jogos e criptografar toda a comunicação cliente/servidor.
- Grupo de métricas – O grupo usado para agregar métricas para várias frotas.
- Política de proteção de escalabilidade de jogos –Configuração atual para [proteção de sessão de jogo](#) para a frota.
- Máximo de sessões de jogo por jogador – O número máximo de sessões que um jogador pode criar durante o período da Política.
- Período da política – Quanto tempo esperar até redefinir o número de sessões que um jogador criou.

Detalhes da compilação

A seção Detalhes da compilação exibe a compilação hospedada na frota. Selecione o nome da compilação para visualizar a página de detalhes da compilação completa.

Configuração de runtime

A seção Configuração do runtime exibe os processos do servidor a serem executados em cada instância. Ela inclui o caminho do executável do servidor de jogos e os parâmetros de inicialização opcionais.

Ativação da sessão do jogo

A seção Ativação da sessão de jogo exibe o número de processos do servidor que são iniciados ao mesmo tempo e quanto tempo esperar até que o processo seja ativado antes de encerrá-lo.

EC2 configurações de porta

A seção Portas exibe as permissões de conexão da frota, inclusive os intervalos de configuração do endereço IP e da porta.

Métricas

O guia Métricas exibe uma representação gráfica das métricas da frota ao longo do tempo. Para obter mais informações sobre o uso de métricas em Amazon GameLift Servers, consulte [Monitore Amazon GameLift Servers com a Amazon CloudWatch](#).

Eventos

O guia Eventos fornece um log de todos os eventos ocorridos na frota, inclusive o código do evento, a mensagem e o carimbo de data/hora. Veja as descrições dos [eventos](#) no Amazon GameLift Servers Referência de API.

Escalabilidade

O guia Escalabilidade contém informações sobre a capacidade da frota, inclusive o status atual e as alterações de capacidade ao longo do tempo. Ela também fornece ferramentas para atualizar os limites de capacidade e gerenciar o ajuste de escala automático.

Escalabilidade da capacidade

Veja as configurações atuais de capacidade da frota para cada localização da frota. Para obter mais informações sobre como alterar limites e capacidade, consulte [Escalando a capacidade de hospedagem de jogos com Amazon GameLift Servers](#).

- AWS Local – Nome de um local em que as instâncias da frota são implantadas.
- Status – Status de hospedagem da localização da frota. O status da localização deve ser ACTIVE para poder hospedar jogos.
- Tamanho mínimo – O menor número de instâncias que devem ser implantadas no local.
- Instâncias desejadas – O número alvo de instâncias ativas para manter a localização. Quando as instâncias ativas e as desejadas não são as mesmas, um evento de escalabilidade é iniciado para iniciar ou encerrar as instâncias conforme necessário até que as instâncias ativas sejam iguais às desejadas.
- Tamanho máximo – o maior número de instâncias que podem ser implantadas no local.
- Disponível – O limite de serviço nas instâncias menos o número de instâncias em uso. Esse valor informa o número máximo de instâncias do que você poderá adicionar ao local.

Políticas de ajuste de escala automático

Esta seção aborda as informações sobre as políticas de ajuste de escala automático que são aplicadas à frota. Configure ou atualize uma política com base em destino. As políticas baseadas em regras da frota, que devem ser definidas usando o AWS SDK ou a CLI, são exibidas aqui. Para ter mais informações sobre a escalabilidade, consulte [Capacidade de frota de escala automática com Amazon GameLift Servers](#).

Histórico de escalabilidade

Visualize gráficos das mudanças de capacidade ao longo do tempo.

Localizações

O guia Localizações lista todos os locais em que as instâncias da frota são implantadas. Os locais incluem a região de origem da frota e quaisquer locais remotos que tenham sido adicionados. É possível adicionar ou remover locais diretamente nessa guia.

- Local – Nome de um local em que as instâncias da frota são implantadas.
- Status – Status de hospedagem da localização da frota. O status do local rastreia o processo de ativação das primeiras instâncias no local. O status da localização deve ser ACTIVE para poder hospedar jogos.
- Instâncias ativas – O número de instâncias com processos de servidor em execução no local da frota.
- Servidores ativos – O número de processos do servidor de jogos com possibilidade de hospedar sessões de jogo no local da frota.
- Sessões de jogos – O número de sessões de jogos ativas em instâncias no local da frota.
- Sessões de jogadores – O número de sessões de jogadores, que representam jogadores individuais, que estão participando de sessões de jogo ativas na localização da frota.

Sessões de jogos

O guia Sessões de jogos lista as sessões de jogos passadas e atuais hospedadas na frota, incluindo algumas informações detalhadas. Selecione o ID de uma sessão do jogo para acessar informações adicionais da sessão de jogos, inclusive sessões de jogadores. Para obter mais informações sobre sessões de jogadores, consulte [Sessões de jogos e jogadores no Amazon GameLift Servers console](#).

Sessões de jogos e jogadores no Amazon GameLift Servers console

Você pode usar o Amazon GameLift Servers console para trabalhar com sessões de jogos e sessões de jogadores. Para obter mais informações sobre sessões de jogos e sessões de jogadores, consulte [Como os jogadores se conectam aos jogos](#). O Amazon GameLift Servers console fornece informações e ferramentas para ajudá-lo a investigar problemas com suas sessões de jogo.

O que você pode fazer:

- Explore a sessão do jogo e a atividade da sessão do jogador hospedadas em uma frota específica.
- Pesquise a atividade da sessão de jogo de um jogador específico em várias frotas.
- Encerre uma sessão de jogo específica.

Exibir detalhes da sessão do jogo

Os dados da sessão do jogo e da sessão do jogador são organizados pela frota que hospeda a sessão do jogo.

Para acessar as informações da sessão do jogo e da sessão do jogador

1. No [Amazon GameLift Servers console](#), abra o painel de navegação esquerdo. Selecione um tipo de solução de hospedagem e abra a página Fleets. Por exemplo:
 - Hospedagem, em qualquer lugar, frotas
 - Hospedagem, gerenciamento EC2, frotas
 - Hospedagem, contêineres gerenciados, frotas
2. Cada página de frotas exibe a lista de frotas selecionadas atualmente. Região da AWS Escolha a frota da qual você deseja visualizar os dados da sessão de jogo.
3. Na página de detalhes da frota, abra a guia Sessões do jogo. Esta aba lista todas as sessões de jogo hospedadas na frota, juntamente com informações resumidas. Você pode ajustar o conteúdo da tabela conforme necessário usando a ferramenta Preferências (veja o ícone  no canto superior direito da tabela). As preferências personalizadas são salvas no usuário da sua conta AWS e aplicadas automaticamente sempre que você visualizar esta página.
4. Escolha uma sessão de jogo na lista para ver informações adicionais.
5. Se a sessão do jogo incluir dados da sessão do jogador, escolha Exibir sessões do jogador para abrir a ferramenta de pesquisa de sessões do jogador com o ID da sessão do jogo preenchido automaticamente.

Os detalhes das sessões do jogo incluem as seguintes informações:

- Status – Status da sessão de jogo.
 - Ativando – A instância está iniciando uma sessão de jogo.

- Ativo – Uma sessão de jogo está em execução e disponível para receber jogadores, dependendo da [política de criação de jogadores](#) da sessão.
- Encerrada – A sessão de jogo foi encerrada.
- ARN – O nome do recurso da Amazon da sessão do jogo.
- Nome – Nome gerado para a sessão de jogo.
- Local — O local em que a sessão do jogo foi Amazon GameLift Servers hospedada.
- Hora de criação — Data e hora em que a sessão de transmissão Amazon GameLift Servers foi criada.
- Hora de término – Data e hora em que a sessão do jogo terminou.
- Nome DNS – O nome do host da sessão do jogo.
- Endereço IP – Endereço IP especificado para a sessão de jogo.
- Porta – Número de porta usado para se conectar à sessão de jogo.
- ID do criador – Um identificador exclusivo do jogador que iniciou a sessão do jogo.
- Política de criação de sessão de jogador – Indica se a sessão do jogo está aceitando novos jogadores.
- Política de proteção de escalabilidade de jogos — O tipo de proteção de sessão de jogo a ser definida em todas as novas instâncias que Amazon GameLift Servers começam na frota.

Dados de jogo

Dados de propriedades do jogo, formatados como uma string, a serem enviados à sessão de jogo no início.

Propriedades de jogo

Dados de propriedades do jogo, formatados como key/value pares, para enviar para sua sessão de jogo no início.

Dados de marcação de jogos

Se a sessão de jogo foi criada com FlexMatch, os dados de matchmaking descrevem informações sobre a configuração de matchmaking e o conjunto de regras. Isso inclui os recursos dos jogadores e as atribuições da equipe de cada partida. Os dados são no formato JSON. Para obter mais informações sobre FlexMatch matchmaking, consulte [Construir um matchmaker](#).

Pesquisar dados da sessão do jogador

Se sua solução de hospedagem de jogos usa sessões de jogadores e fornece um jogador exclusivo IDs, você pode explorar atividades específicas do jogador em sessões de jogos anteriores ou atuais em várias frotas. Abra a ferramenta de pesquisa de sessão do Player usando um dos seguintes métodos:

- No Amazon GameLift Servers console, abra o painel de navegação esquerdo, escolha Pesquisa de sessão do jogador e selecione o tipo de filtro a ser usado.
- Ao visualizar os detalhes da sessão de jogo de uma frota, escolha Exibir sessões de jogadores. A ferramenta de pesquisa é aberta com a sessão do jogo com o filtro de ID da sessão do jogo pré-selecionado e o valor da sessão do jogo preenchido.

Ao usar a ferramenta de pesquisa, você pode fornecer as seguintes informações:

- Um ID da sessão do jogador para obter informações sobre uma sessão específica do jogador.
- Um ID de sessão de jogo para obter informações sobre todas as sessões de jogadores da sessão de jogo solicitada. Os resultados representam todos os jogadores que reservaram um slot ou se conectaram à sessão do jogo. Opcionalmente, você pode filtrar os resultados pelo status da sessão do jogador.
- Um ID de jogador para obter informações sobre todas as sessões do jogador solicitado. Os resultados representam todas as sessões de jogo das quais o jogador participou.

Note

A ferramenta de pesquisa pesquisa todas as atividades da sessão do jogador nas atualmente selecionadas Região da AWS. Se você tiver várias frotas na região, os resultados incluem atividades de sessões de jogadores em todas as frotas. Para frotas com vários locais, os resultados também incluem atividades de sessões de jogadores nas localizações remotas das frotas.

Os seguintes dados da sessão de jogador são coletados para cada sessão de jogo:

- ID da sessão do jogador – O identificador atribuído à sessão do jogador.

- ID de jogador – Um identificador exclusivo para o jogador. Selecione nesse ID para obter informações adicionais sobre o jogador.
- ID da sessão do jogo: o identificador atribuído à sessão do jogo.
- ID da frota: o identificador atribuído à frota que hospedou a sessão do jogo.
- Status – O status da sessão de jogador. A seguir estão os status possíveis:
 - Reservado – A sessão de jogador foi reservada, mas os jogadores não estão conectados.
 - Ativo) – A sessão de jogador está conectada atualmente ao servidor de jogos.
 - Concluíd – A sessão de jogador foi encerrada; o jogador não está mais conectado.
 - Tempo limite – O jogador não conseguiu se conectar.
- CHora de início – A hora em que o jogador se conectou à sessão de jogo.
- Hora de término – A hora em que o jogador se desconectou da sessão de jogo.
- Dados de conexão: endereço IP, nome DNS e porta que o jogador usou para se conectar à sessão do jogo.
- Dados do jogador: informações sobre o jogador fornecidas durante a criação da sessão do jogador.

Encerrar uma sessão de jogo

Use o Amazon GameLift Servers console para encerrar uma sessão de jogo específica. Esse recurso oferece um método simples e rápido para localizar uma sessão de jogo e enviar um sinal para encerrá-la. Outro método de encerramento exige que você encontre a instância da frota em que a sessão do jogo está sendo executada, acesse remotamente a instância e encerre manualmente a sessão do jogo.

Você pode encerrar uma sessão de jogo por qualquer motivo. O motivo mais comum é resolver uma sessão de jogo que não está sendo encerrada naturalmente. Como resultado, o recurso de hospedagem da sessão de jogo não pode ser liberado para hospedar uma nova sessão de jogo, e a capacidade de hospedagem da frota é reduzida.

Note

Esse recurso depende de determinadas configurações para sua solução de hospedagem. Ele tem as seguintes limitações:

- A sessão do jogo deve ser hospedada em uma frota que esteja executando um servidor de jogos criado com SDK de servidor para Amazon GameLift Servers v5 ou superior. Se seus

servidores de jogos forem implantados com uma versão mais antiga, você precisará usar o acesso remoto para excluir a sessão do jogo.

- Se a sessão de jogo for hospedada em uma frota Anywhere, a frota deverá estar usando o Amazon GameLift Servers Agente para gerenciar os processos do servidor de jogos.

Para encerrar uma sessão de jogo

1. No [Amazon GameLift Servers console](#), abra o painel de navegação esquerdo. Selecione um tipo de solução de hospedagem e abra a página Fleets. Por exemplo:
 - Hospedagem, em qualquer lugar, frotas
 - Hospedagem, gerenciamento EC2, frotas
 - Hospedagem, contêineres gerenciados, frotas
2. Cada página de frotas exibe a lista de frotas selecionadas atualmente. Região da AWS Escolha a frota que está hospedando a sessão de jogo que você deseja encerrar.
3. Na página de detalhes da frota, abra a guia Sessões do jogo. Na lista de sessões de jogo, selecione aquela que você deseja encerrar e escolha o botão Encerrar.
4. Na sessão de encerramento do jogo? janela, verifique se você está encerrando a sessão de jogo correta e escolha um método de encerramento.
 - Desligamento normal da sessão de jogo — Essa opção envia um sinal para que o processo do servidor que está hospedando a sessão de jogo seja encerrado. Se a compilação do seu servidor de jogos foi devidamente integrada Amazon GameLift Servers, o processo do servidor inicia a sequência de encerramento da sessão de jogo, notifica Amazon GameLift Servers que ela está terminando e para. Dependendo do design do jogo, a sequência de desligamento pode incluir etapas para concluir a sessão de jogo normalmente, como salvar dados e notificar jogadores ativos. Esse método pode exigir um pequeno atraso para concluir a sequência de encerramento da sessão do jogo.
 - Desligamento imediato da sessão de jogo — Essa opção envia um sinal ao gerenciador de processos para encerrar o processo do servidor que está hospedando a sessão do jogo. Essa opção ignora o encerramento normal da sessão de jogo. Ele é capaz de encerrar a sessão do jogo mesmo quando o processo do servidor não consegue responder.
5. Confirme o encerramento da sessão do jogo. Você pode acompanhar o progresso do desligamento na página do console de sessões do jogo. O status da sessão do jogo mudará para “Encerrada” e depois para “Encerrada” quando o desligamento for concluído.

Tópicos relacionados

- Você também pode encerrar as sessões de jogo usando o AWS SDK e o AWS CLI. Para obter mais detalhes e exemplos, consulte o tópico de referência Amazon GameLift Servers da API [TerminateGameSession](#).
- Para obter mais informações sobre a integração do servidor de jogos e como um processo do servidor responde aos sinais do Amazon GameLift Servers serviço, consulte [Adicione Amazon GameLift Servers ao seu servidor de jogos](#).

Aliases no Amazon GameLift Servers console

A página Alias exibe informações sobre Amazon GameLift Servers aliases que direcionam destinos de hospedagem específicos do tráfego. Para aliases, escolha Hosting, Aliases no painel de navegação.

Na página de aliases, é possível fazer o seguinte:

- Crie um alias. Escolha Criar alias.
- Filtre e classifique a lista de tabelas. Use os controles na parte superior da tabela. Você pode ajustar o conteúdo da tabela conforme necessário usando a ferramenta Preferências (veja o ícone  no canto superior direito da tabela). As preferências personalizadas são salvas no usuário da sua conta AWS e aplicadas automaticamente sempre que você visualizar esta página.
- Visualize detalhes do alias. Selecione um nome de alias para abrir a página de detalhes do alias.
- Exclua um alias. Selecione um alias e, em seguida, selecione Excluir.

Detalhes do alias

A página de detalhes do alias exibe informações sobre o alias.

Nesta página, você poderá:

- Editar um alias. Selecione a opção Editar.
- Visualize as frotas que você associou ao alias.
- Exclua um alias. Selecione a opção Excluir.

Entre as informações detalhadas do alias estão:

- ID – O número exclusivo usado para identificar o alias.
- Descrição – A descrição do alias.
- ARN – O nome do recurso da Amazon do alarme do alias.
- Criação – A data e a hora em que o alias foi criado.
- Última atualização – A data e hora em que o alias foi atualizado pela última vez.
- Tipo de roteamento – O tipo de opção de roteamento para o alias, que pode ser um destes:
 - Simples – Roteia o tráfego do jogador para um ID de frota especificado. Você pode atualizar o ID da frota para um alias a qualquer momento.
 - Terminal – Reenvia uma mensagem para o cliente. Por exemplo, você pode direcionar jogadores que estão usando um out-of-date cliente para um local onde eles possam obter um upgrade.
- Tags – pares de chave e valor usados para identificar o alias.

Filas de sessões de jogo no Amazon GameLift Servers console

Você pode ver informações sobre todas as filas, que são usadas para processar solicitações de novas sessões de jogo. A página de filas mostra as filas de sessões de jogo no momento selecionado. Região da AWS Na Queues, você pode criar uma nova fila, excluir filas existentes ou abrir uma página de detalhes de uma fila selecionada. Cada página de detalhes da fila contém os dados de configuração e métricas da fila. Para obter mais informações sobre filas, consulte [Gerenciando o posicionamento da sessão de jogo com Amazon GameLift Servers filas](#).

A página de filas exibe as informações de resumo a seguir para cada fila. Você pode ajustar o conteúdo da tabela conforme necessário usando a ferramenta Preferências (veja o ícone



no canto superior direito da tabela). As preferências personalizadas são salvas no usuário da sua conta AWS e aplicadas automaticamente sempre que você visualizar esta página.

- Nome da fila – O nome atribuído à fila. As solicitações de novas sessões de jogos especificam uma fila por esse nome.
- Tempo limite da fila – Tempo máximo, em segundos, que uma solicitação de colocação de sessão de jogo permanece na fila antes do tempo limite.
- Destinos na fila – Número de frotas listadas na configuração da fila. Amazon GameLift Servers coloca novas sessões de jogo em qualquer frota na fila.

Visualizar detalhes da fila

Você pode acessar informações detalhadas sobre qualquer fila, inclusive a configuração e as métricas da fila. Para abrir uma página de detalhes da fila, acesse a página Filas e escolha um nome de fila.

A página de detalhes da fila exibe uma tabela de resumo e guias contendo informações adicionais. Nesta página, você pode fazer o seguinte:

- Atualize a configuração da fila, a lista de destinos e as políticas de latência de jogador. Selecione a opção Editar.
- Excluir uma fila. Depois de excluir uma fila, todas as solicitações de novas sessões de jogos que referenciem o nome dessa fila falharão. Selecione a opção Excluir.

Note

Para restaurar uma fila excluída, crie uma nova fila com o nome da fila excluída.

Detalhes

Visão geral

A seção Visão geral exibe o nome de recurso da fila e o nome do recurso da Amazon (ARN) da fila e o Tempo limite. Você pode usar o ARN ao referenciar a fila em outras ações ou áreas do Amazon GameLift Servers. O tempo limite é o tempo máximo, em segundos, durante o qual uma solicitação de colocação de sessão de jogo permanece na fila antes de atingir o tempo limite.

Notificação de eventos

A seção Notificação de eventos lista o tópico do SNS. Amazon GameLift Servers publica notificações de eventos e os dados de eventos que são adicionados a todos os eventos criados por essa fila.

Tags

A tabela Tags exibe as chaves e os valores usados para marcar o recurso. Para obter mais informações sobre marcação, consulte Recursos de [marcação AWS](#).

Métricas

O guia Métricas mostra uma representação gráfica das métricas de fila com o passar do tempo.

As métricas de fila incluem uma variedade de informações que descrevem a atividade de posicionamento na fila, inclusive posicionamentos bem-sucedidos organizados por região. Você poderá usar os dados da região para entender onde você está hospedando seus jogos. As métricas de posicionamento regional podem ajudar a detectar problemas com o design geral da fila.

As métricas de fila também estão disponíveis na Amazon CloudWatch. Para obter descrições das métricas disponíveis, consulte [Métricas do Amazon GameLift Servers para filas](#).

Destinos

O guia Destinos mostra todas as frotas ou aliases listados para a fila.

Quando Amazon GameLift Servers pesquisa nos destinos os recursos disponíveis para hospedar uma nova sessão de jogo, ele pesquisa na ordem padrão listada aqui. Desde que haja capacidade no primeiro destino listado, Amazon GameLift Servers coloca novas sessões de jogo lá. As solicitações de colocação de sessões de jogos individuais podem substituir a ordem padrão quando você fornece dados de latência para o jogador. Esses dados dizem Amazon GameLift Servers para pesquisar um destino disponível com a menor latência média do jogador. Para obter mais informações sobre como projetar suas filas, consulte [Personalize uma fila de sessões de jogo](#).

Posicionamento da sessão

Políticas de latência de jogador

A seção Políticas de latência do player mostra todas as políticas que a fila usa. As tabelas listam as políticas na ordem em que são aplicadas.

Locais

A seção Locais mostra os locais em que essa fila pode colocar uma sessão de jogo.

Prioridade

A seção Prioridade mostra a ordem em que a fila avalia os detalhes de uma sessão de jogo.

Ordem de local

A seção Ordem de localização mostra a ordem padrão que a fila usa ao colocar sessões de jogo. A fila usa essa ordem se você não tiver definido outros tipos de prioridade.

Monitore Amazon GameLift Servers com a Amazon CloudWatch

Você pode monitorar Amazon GameLift Servers usando a Amazon CloudWatch, um AWS serviço que coleta dados brutos e os processa em métricas legíveis, quase em tempo real. Essas estatísticas são mantidas por 15 meses para fornecer uma perspectiva histórica sobre o desempenho do seu servidor de jogos com o Amazon GameLift Servers. É possível definir alarmes que observam determinados limites e enviam notificações ou realizam ações quando esses limites são atingidos. Para obter mais informações, consulte o [Guia CloudWatch do usuário da Amazon](#).

A tabela a seguir lista as métricas e as dimensões do Amazon GameLift Servers. Todas as métricas disponíveis no também CloudWatch estão disponíveis no Amazon GameLift Servers console, que fornece os dados como um conjunto de gráficos personalizáveis. Para acessar CloudWatch as métricas dos seus jogos, use a AWS Management Console AWS CLI, a ou a CloudWatch API.

Se uma métrica não tiver uma localização, ela usará a localização inicial.

Dimensões para métricas do Amazon GameLift Servers

O Amazon GameLift Servers dá suporte à filtragem de métricas pelas dimensões a seguir.

Dimensão	Descrição
Location	Filtre métricas para um local de implantação da frota. Se uma métrica não tiver uma localização, ela usará a localização inicial.
FleetId	Filtre métricas para uma única frota. Essa dimensão pode ser usada com todas as métricas de frotas para instâncias, processos de servidor, sessões de jogos e sessões de jogador.
MetricGroup	Filtre métricas para uma coleção de frotas. Adicione uma frota a um grupo de métricas adicionando o nome do grupo métrico aos atributos da frota (consulte UpdateFleetAttributes()). Essa dimensão pode ser usada com todas as métricas de frotas para instâncias, processos de servidor, sessões de jogos e sessões de jogador.

Dimensão	Descrição
QueueName	Filtre métricas para uma única fila. Essa dimensão é usada apenas com métricas para filas da sessão de jogo.
ConfigurationName	Filtre métricas para uma única configuração de marcação de jogos. Essa dimensão é usada somente com métricas para configurações de marcação de jogos.
ConfigurationName-RuleName	Filtre métricas para uma interseção de uma configuração de marcação de jogos e regra de marcação de jogos. Essa dimensão é usada somente com métricas para regras de marcação de jogos.
InstanceType	Filtre métricas para uma designação de tipo de EC2 instância, como “c4.large”. Essa dimensão é usada somente com métricas para instâncias spot.
OperatingSystem	Filtrar métricas para o sistema operacional de uma instância. Essa dimensão é usada com métricas para instâncias spot.
GameServerGroup	Filtre métricas do FleetIQ para um grupo de servidores de jogos.

Métricas do Amazon GameLift Servers para frotas

O namespace `AWS/GameLift` inclui as métricas a seguir relacionadas a uma atividade em uma frota ou em grupo de frotas. As frotas são usadas com uma solução gerenciada do Amazon GameLift Servers. O Amazon GameLift Servers serviço envia métricas a CloudWatch cada minuto.

Instâncias

Métrica	Descrição
ActiveInstances	Instâncias com status ATIVO, o que significa que estão executando processos de servidores ativos. A contagem inclui instâncias ociosas e aquelas que estão hospedando uma ou mais sessões de jogo. Essa métrica mede a capacidade total da instância atual. Essa métrica pode ser usada com ajuste de escala automático. Unidades: contagem CloudWatch Estatísticas relevantes: Média, Mínimo, Máximo Dimensões: localização
DesiredInstances	Número desejado de instâncias ativas que o Amazon GameLift Servers está trabalhando para manter na frota. Com ajuste de escala automático, esse valor é determinado com base nas políticas de escalabilidade atualmente vigentes. Sem ajuste de escala automático, esse valor é definido manualmente. Essa métrica não está disponível durante a visualização de dados para grupos de métricas da frota. Unidades: contagem CloudWatch Estatísticas relevantes: Média, Mínimo, Máximo Dimensões: localização
IdleInstances	As instâncias ativas que hospedam atualmente zero (0) sessões de jogo. Essa métrica mede a capacidade e disponível, mas não usada. Essa métrica pode ser usada com ajuste de escala automático.

Métrica	Descrição
	Unidades: contagem CloudWatch Estatísticas relevantes: Média, Mínimo, Máximo Dimensões: localização
MaxInstances	Número máximo de instâncias permitidas para a frota. O máximo de instâncias de uma frota determina o teto da capacidade durante o aumento manual ou automático. Essa métrica não está disponível durante a visualização de dados para grupos de métricas da frota. Unidades: contagem CloudWatch Estatísticas relevantes: Média, Mínimo, Máximo Dimensões: localização
MinInstances	Número mínimo de instâncias permitidas para a frota. O mínimo de instâncias de uma frota determina o piso da capacidade durante a redução manual ou automática. Essa métrica não está disponível durante a visualização de dados para grupos de métricas da frota. Unidades: contagem CloudWatch Estatísticas relevantes: Média, Mínimo, Máximo Dimensões: localização

Métrica	Descrição
PercentIdleInstances	Porcentagem de todas as instâncias ativas ociosas (calculadas como <code>IdleInstances / ActiveInstances</code>). Essa métrica pode ser usada para ajuste de escala automático. Unidades: percentual CloudWatch Estatísticas relevantes: Média, Mínimo, Máximo Dimensões: localização
RecycledInstances	Número de instâncias spot que foram recicladas e substituídas. Amazon GameLift Servers recicla instâncias pontuais que não estão hospedando sessões de jogo no momento e têm uma alta probabilidade de interrupção. Unidades: contagem CloudWatch Estatísticas relevantes: soma, média, mínimo, máximo Dimensões: localização
InstanceInterruptions	Número de instâncias spot interrompidas. Unidades: contagem CloudWatch Estatísticas relevantes: soma, média, mínimo, máximo Dimensões: localização

Métrica	Descrição
CPUUtilization	EC2 métrica. Pois Amazon GameLift Servers essa métrica representa o desempenho do hardware em todas as instâncias ativas em um local de frota. A porcentagem do tempo físico de CPU que a Amazon EC2 usa para executar a instância, que inclui o tempo gasto para executar o código do usuário e o EC2 código da Amazon. As ferramentas em seu sistema operacional podem mostrar uma porcentagem em diferente da CloudWatch devida a fatores como simulação de dispositivos legados, configuração de dispositivos não legados, cargas de trabalho com muita interrupção, migração ao vivo e atualização em tempo real. Unidades: percentual
NetworkIn	EC2 métrica. Pois Amazon GameLift Servers essa métrica representa o desempenho do hardware em todas as instâncias ativas em um local de frota. O número de bytes recebidos em todas as interfaces de rede pela instância. Essa métrica identifica o volume de tráfego de rede de entrada para um aplicativo em uma única instância. Unidades: bytes
NetworkOut	EC2 métrica. Pois Amazon GameLift Servers essa métrica representa o desempenho do hardware em todas as instâncias ativas em um local de frota. O número de bytes enviados em todas as interfaces de rede pela instância. Essa métrica identifica o volume de tráfego de rede de saída para um aplicativo em uma única instância. Unidades: bytes

Métrica	Descrição
DiskReadBytes	EC2 métrica. Pois Amazon GameLift Servers essa métrica representa o desempenho do hardware em todas as instâncias ativas em um local de frota. Bytes lidos de todos os volumes de armazenamento de instâncias disponíveis para a instância. Essa métrica é utilizada para determinar o volume de dados que a aplicação lê do disco rígido da instância. Você poderá usá-lo para determinar a velocidade da aplicação. Unidades: bytes
DiskWriteBytes	EC2 métrica. Pois Amazon GameLift Servers essa métrica representa o desempenho do hardware em todas as instâncias ativas em um local de frota. Bytes gravados em todos os volumes de armazenamento de instâncias disponíveis para a instância. Essa métrica é utilizada para determinar o volume de dados que a aplicação grava no disco rígido da instância. Você poderá usá-lo para determinar a velocidade da aplicação. Unidades: bytes
DiskReadOps	EC2 métrica. Pois Amazon GameLift Servers essa métrica representa o desempenho do hardware em todas as instâncias ativas em um local de frota. Operações de leitura concluídas de todos os volumes de armazenamento de instâncias disponíveis para a instância em um período de tempo especificado. Para calcular a média de I/O operações por segundo (IOPS) do período, divida o total de operações no período pelo número de segundos nesse período. Unidades: contagem

Métrica	Descrição
DiskWriteOps	EC2 métrica. Pois Amazon GameLift Servers essa métrica representa o desempenho do hardware em todas as instâncias ativas em um local de frota. Operações de gravação concluídas em todos os volumes de armazenamento de instâncias disponíveis para a instância em um período de tempo especificado. Para calcular a média de I/O operações por segundo (IOPS) do período, divida o total de operações no período pelo número de segundos nesse período. Unidades: contagem

Processos de servidor

Métrica	Descrição
ActiveServerProcesses	Processos de servidor com status ATIVO, o que significa que estão executando e podem hospedar sessões de jogo. A contagem inclui processos de servidor ociosos e aqueles que estão hospedando sessões de jogo. Essa métrica mede a capacidade total do processo de servidor. Unidades: contagem CloudWatch Estatísticas relevantes: Média, Mínimo, Máximo Dimensões: localização
HealthyServerProcesses	Processos de servidores ativos informados como íntegros. Essa métrica é útil para acompanhar a integridade geral dos servidores de jogo da frota. Unidades: contagem

Métrica	Descrição
	CloudWatch Estatísticas relevantes: Média, Mínimo, Máximo Dimensões: localização
PercentHealthyServerProcesses	Porcentagem de todos os processos de servidor ativos informados como íntegros (calculada como $\text{HealthyServerProcesses} / \text{ActiveServerProcesses}$). Unidades: percentual CloudWatch Estatísticas relevantes: Média, Mínimo, Máximo Dimensões: localização
ServerProcessAbnormalTerminations	Processos de servidor que foram desligados por causa de circunstâncias anormais desde o relatório mais recente. Essa métrica inclui encerramentos que foram iniciados pelo serviço Amazon GameLift Servers. Isso ocorre quando um processo do servidor para de responder, relata consistentemente falhas nas verificações de integridade ou não é encerrado de forma limpa (chamando () ProcessEnding). Unidades: contagem CloudWatch Estatísticas relevantes: soma, média, mínimo, máximo Dimensões: localização

Métrica	Descrição
ServerProcessActivations	Processos de servidor transicionados com êxito do status ACTIVATING para ACTIVE desde o relatório mais recente. Os processos de servidor não podem hospedar sessões de jogo até que estejam ativos. Unidades: contagem CloudWatch Estatísticas relevantes: soma, média, mínimo, máximo Dimensões: localização
ServerProcessTerminations	Processos de servidor que foram desligados desde o relatório mais recente. Isso inclui todos os processos de servidor transicionados para o status TERMINATED por qualquer motivo, inclusive encerramentos de processo normais e anormais. Unidades: contagem CloudWatch Estatísticas relevantes: soma, média, mínimo, máximo Dimensões: localização

Sessões de jogos

Métrica	Descrição
ActivatingGameSessions	Sessões de jogo com status ACTIVATING, o que significa que estão no processo de inicialização. As sessões de jogo não podem hospedar jogadores até que estejam ativos. Números altos por um período sustentado podem indicar que as sessões de jogo não estão fazendo a transição do status ACTIVATING para o status ACTIVE.

Métrica	Descrição
	G para ACTIVE. Essa métrica pode ser usada com ajuste de escala automático. Unidades: contagem CloudWatch Estatísticas relevantes: Média, Mínimo, Máximo Dimensões: localização
ActiveGameSessions	Sessões de jogo com status ACTIVE, o que significa que eles podem hospedar jogadores e estão hospedando zero ou mais jogadores. Essa métrica mede o número total de sessões de jogo hospedados no momento. Essa métrica pode ser usada com ajuste de escala automático. Unidades: contagem CloudWatch Estatísticas relevantes: Média, Mínimo, Máximo Dimensões: localização

Métrica	Descrição
AvailableGameSessions	Processos de servidor ativos e íntegros que não estão sendo usados atualmente para hospedar uma sessão de jogo e podem iniciar uma nova sessão de jogo sem demora para ativar novos processos ou instâncias do servidor. Essa métrica pode ser usada com ajuste de escala automático.  Note Para frotas que limitam as ativações simultâneas de sessões de jogo, use a métrica <code>ConcurrentActiveGameSessions</code>. Essa métrica representa com mais precisão o número de novas sessões de jogo que podem começar sem nenhum tipo de atraso. Unidades: contagem CloudWatch Estatísticas relevantes: Média, Mínimo, Máximo Dimensões: localização

Métrica	Descrição
<code>ConcurrentActivatableGameSessions</code>	Processos de servidor ativos e saudáveis que não estão sendo usados atualmente para hospedar uma sessão de jogo e podem iniciar imediatamente uma nova sessão de jogo. Essa métrica difere da <code>AvailableGameSessions</code> na seguinte forma: ela não conta os processos do servidor que atualmente não podem ativar uma nova sessão de jogo devido aos limites nas ativações da sessão de jogo. (Veja a configuração RuntimeConfiguration opcional da frota <code>MaxConcurrentGameSessionActivations</code>). Para frotas que não limitam as ativações de sessões de jogo, essa métrica é idêntica a <code>AvailableGameSessions</code> . Unidades: contagem CloudWatch Estatísticas relevantes: Média, Mínimo, Máximo Dimensões: localização
<code>PercentAvailableGameSessions</code>	Porcentagem de slots de sessão de jogo em todos os processos de servidor ativos (íntegros ou não) que não estão sendo usados no momento (calculados como $\text{AvailableGameSessions} / [\text{ActiveGameSessions} + \text{AvailableGameSessions} + \text{unhealthy server processes}]$). Essa métrica pode ser usada com ajuste de escala automático. Unidades: percentual CloudWatch Estatísticas relevantes: Média Dimensões: localização

Métrica	Descrição
GameSessionInterruptions	Número de sessões de jogos em instâncias spot interrompidas. Unidades: contagem CloudWatch Estatísticas relevantes: soma, média, mínimo, máximo Dimensões: localização

Sessões de jogador

Métrica	Descrição
CurrentPlayerSessions	Sessões de jogador com status ACTIVE (o jogador está conectado a uma sessão de jogo ativa) ou status RESERVED (o jogador recebeu um slot em uma sessão de jogo, mas ainda não se conectou). Essa métrica pode ser usada com ajuste de escala automático. Unidades: contagem CloudWatch Estatísticas relevantes: Média, Mínimo, Máximo
PlayerSessionActivations	Sessões de jogador transicionados do status RESERVED para ACTIVE desde o relatório mais recente. Isso ocorre quando um jogador se conecta com êxito a uma sessão de jogo ativa. Unidades: contagem CloudWatch Estatísticas relevantes: soma, média, mínimo, máximo

Métricas do Amazon GameLift Servers para filas

O namespace Amazon GameLift inclui as métricas a seguir relacionadas a uma atividade em uma fila de colocação da sessão de jogo. As filas são usadas com uma solução gerenciada do Amazon GameLift Servers. O Amazon GameLift Servers serviço envia métricas a CloudWatch cada minuto.

Métrica	Descrição
AverageWaitTime	Tempo médio em que as solicitações de colocação da sessão de jogo na fila com o status PENDING esperaram ser atendidas. Unidades: segundos CloudWatch Estatísticas relevantes: Média, Mínimo, Máximo, Soma Dimensões: localização
FirstChoiceNotViable	As sessões de jogo que foram posicionadas com êxito, mas NÃO na primeira opção frota, pois esta foi considerada inviável (como uma frota spot com uma alta taxa de interrupção). Essa métrica é baseada no custo, não na latência. A frota de primeira escolha é a primeira frota listada na fila ou, quando uma solicitação de colocação inclui dados de latência do jogador, é a primeira frota escolhida por priorização. FleetIQ Se não houver frotas spot viáveis, qualquer frota na região podem ser selecionada. Unidades: contagem CloudWatch Estatísticas relevantes: Média, Mínimo, Máximo, Soma
FirstChoiceOutOfCapacity	As sessões de jogo que foram posicionadas com êxito, mas NÃO na primeira opção frota, pois esta não tinha recursos disponíveis. A frota de primeira escolha é a primeira frota listada na fila ou, quando

Métrica	Descrição
	uma solicitação de colocação inclui dados de latência do jogador, é a primeira frota escolhida de acordo com sua priorização definida. FleetIQ Unidades: contagem CloudWatch Estatísticas relevantes: Média, Mínimo, Máximo, Soma
LowestLatencyPlacement	As sessões de jogo que foram posicionadas com êxito em uma região que oferece a latência mais baixa possível da fila para os jogadores. Essa métrica é emitida apenas quando dados de latência do jogador são incluídos na solicitação de posicionamento. Unidades: contagem CloudWatch Estatísticas relevantes: Média, Mínimo, Máximo, Soma
LowestPricePlacement	As sessões de jogo que foram posicionadas com êxito em uma frota com o preço mais baixo possível da fila para a região selecionada. Essa frota poderá ser uma frota spot ou uma instância sob demanda se a fila não tiver instâncias spot. Essa métrica é emitida apenas quando dados de latência do jogador são incluídos na solicitação de posicionamento. Unidades: contagem CloudWatch Estatísticas relevantes: Média, Mínimo, Máximo, Soma

Métrica	Descrição
Placement <region name>	As sessões de jogo que foram posicionadas com êxito em frotas localizadas na região especificada. Essa métrica divide a métrica PlacementSucceeded por região. Unidades: contagem CloudWatch Estatísticas relevantes: soma
PlacementsCanceled	Solicitações de colocação da sessão de jogo que foram canceladas antes do tempo limite desde o último relatório. Unidades: contagem CloudWatch Estatísticas relevantes: Média, Mínimo, Máximo, Soma
PlacementsFailed	As solicitações de posicionamento da sessão de jogo que por qualquer motivo falharam desde o último relatório. Unidades: contagem CloudWatch Estatísticas relevantes: Média, Mínimo, Máximo, Soma
PlacementsStarted	Novas solicitações de posicionamento da sessão de jogo que foram adicionadas à fila desde o último relatório. Unidades: contagem CloudWatch Estatísticas relevantes: Média, Mínimo, Máximo, Soma

Métrica	Descrição
PlacementsSucceeded	Solicitações de posicionamento da sessão de jogo que resultaram em uma nova sessão de jogo desde o último relatório. Unidades: contagem CloudWatch Estatísticas relevantes: Média, Mínimo, Máximo, Soma
PlacementsTimedOut	Solicitações de posicionamento da sessão de jogo que alcançaram o tempo limite da fila sem serem atendidas desde o último relatório. Unidades: contagem CloudWatch Estatísticas relevantes: Média, Mínimo, Máximo, Soma
QueueDepth	Número de solicitações de posicionamento de sessão de jogo na fila com status PENDING. Unidades: contagem CloudWatch Estatísticas relevantes: Média, Mínimo, Máximo, Soma Dimensões: localização

Métricas do Amazon GameLift Servers para marcação de jogos

O namespace `Amazon GameLift Servers` inclui métricas sobre atividades do FlexMatch para configurações de marcação de jogos e regras de marcação de jogos. A marcação de jogos do FlexMatch é usada com uma solução gerenciada do Amazon GameLift Servers. O Amazon GameLift Servers serviço envia métricas a CloudWatch cada minuto.

Para obter mais informações sobre a sequência da atividade de matchmaking, veja [Como Amazon GameLift Servers FlexMatch funciona](#).

Configurações de marcação de jogos

Métrica	Descrição
CurrentTickets	As solicitações de marcação de jogos que estão sendo processadas no momento ou aguardando para serem processadas. Unidades: contagem CloudWatch Estatísticas relevantes: Média, Mínimo, Máximo, Soma
MatchAcceptancesTimedOut	Para as configurações de marcação de jogos que exigem aceitação, as correspondências em potencial que expiraram durante a aceitação desde o último relatório. Unidades: contagem CloudWatch Estatísticas relevantes: soma
MatchesAccepted	Para as configurações de marcação de jogos que exigem aceitação, as correspondências em potencial que foram aceitas desde o último relatório. Unidades: contagem CloudWatch Estatísticas relevantes: soma
MatchesCreated	Jogos em potencial que foram criados desde o último relatório. Unidades: contagem CloudWatch Estatísticas relevantes: soma
MatchesPlaced	Jogos que foram colocados com êxito em uma sessão de jogos desde o último relatório. Unidades: contagem

Métrica	Descrição
	CloudWatch Estatísticas relevantes: soma
MatchesRejected	Para as configurações de marcação de jogos que exigem aceitação, as correspondências em potencial que foram rejeitadas por pelo menos um jogador desde o último relatório. Unidades: contagem CloudWatch Estatísticas relevantes: soma
PlayersStarted	Jogadores em tíquetes de marcação de jogos que foram adicionados desde o último relatório. Unidades: contagem CloudWatch Estatísticas relevantes: soma
TicketsFailed	As solicitações de marcação de jogos que resultaram em uma falha desde o último relatório. Unidades: contagem CloudWatch Estatísticas relevantes: soma
TicketsStarted	As novas solicitações de marcação de jogos que foram criadas desde o último relatório. Unidades: contagem CloudWatch Estatísticas relevantes: soma
TicketsTimedOut	As solicitações de marcação de jogos que atingiram o limite de tempo de espera desde o último relatório. Unidades: contagem CloudWatch Estatísticas relevantes: soma

Métrica	Descrição
TimeToMatch	Para solicitações de marcação de jogos que foram colocadas em um jogo em potencial antes do relatório, o tempo entre a criação do tíquete e a criação do jogo em potencial. Unidades: segundos CloudWatch Estatísticas relevantes: amostras de dados, média, mínimo, máximo
TimeToTicketCancel	Para as solicitações de marcação de jogos que foram canceladas antes do último relatório, o período de tempo entre a criação e o cancelamento do tíquete. Unidades: segundos CloudWatch Estatísticas relevantes: amostras de dados, média, mínimo, máximo
TimeToTicketSuccess	Para as solicitações de marcação de jogos que foram bem-sucedidas antes do último relatório, o período de tempo entre a criação e a disposição de jogos bem-sucedidas. Unidades: segundos CloudWatch Estatísticas relevantes: amostras de dados, média, mínimo, máximo

Regras de marcação de jogos

Métrica	Descrição
RuleEvaluationsPassed	Avaliações da regra durante o processo de marcação de jogos que passaram desde o último

Métrica	Descrição
	relatório. Essa métrica é limitada às 50 regras principais. Unidades: contagem CloudWatchEstatísticas relevantes: soma
RuleEvaluationsFailed	Avaliações da regra durante a marcação de jogos que falharam desde o último relatório. Essa métrica é limitada às 50 regras principais. Unidades: contagem CloudWatch Estatísticas relevantes: soma

Amazon GameLift ServersMétricas do para FleetIQ

O namespace Amazon GameLift Servers inclui métricas para o grupo de servidores de jogos do FleetIQ e a atividade do servidor de jogos como parte de uma solução autônoma do FleetIQ para hospedagem de jogos. O Amazon GameLift Servers serviço envia métricas a CloudWatch cada minuto. Consulte também [Monitorando seus grupos e instâncias do Auto Scaling usando a Amazon CloudWatch no Guia](#) do usuário do Amazon Auto EC2 Scaling.

Métrica	Descrição
AvailableGameServers	Servidores de jogos que estão disponíveis para iniciar uma execução de jogo e não estão ocupados atualmente com uma partida. Esse número inclui servidores de jogos que foram reivindicados, mas ainda estão com status AVAILABLE. Unidades: contagem CloudWatch Estatísticas relevantes da Amazon: soma Dimensões: GameServerGroup

Métrica	Descrição
UtilizedGameServers	Servidores de jogos que estão ocupados com uma partida no momento. Esse número inclui servidores de jogos que estão no status UTILIZED. Unidades: contagem CloudWatch Estatísticas relevantes da Amazon: soma Dimensões: GameServerGroup
DrainingAvailableGameServers	Servidores de jogos em instâncias agendadas para encerramento que não oferecem suporte à jogabilidade no momento. Esses servidores de jogos são a prioridade mais baixa a ser reivindicada em resposta a uma nova solicitação de reivindicação. Unidades: contagem CloudWatch Estatísticas relevantes da Amazon: soma Dimensões: GameServerGroup
DrainingUtilizedGameServers	Servidores de jogos em instâncias agendadas para encerramento que estão oferecendo suporte à jogabilidade no momento. Unidades: contagem CloudWatch Estatísticas relevantes da Amazon: soma Dimensões: GameServerGroup

Métrica	Descrição
PercentUtilizedGameServers	Parte dos servidores de jogos que estão oferecendo o suporte a execuções de jogos no momento. Essa métrica indica a quantidade de capacidade do servidor de jogos que está em uso no momento. Ela é útil para gerar uma política de ajuste de escala automático que pode adicionar e remover instâncias dinamicamente para que correspondam à demanda de jogadores. Unidades: percentual CloudWatch Estatísticas relevantes da Amazon: média, mínimo, máximo Dimensões: GameServerGroup
GameServerInterruptions	Servidores de jogos em instâncias spot que foram interrompidos devido à disponibilidade limitada do spot. Unidades: contagem CloudWatch Estatísticas relevantes da Amazon: soma Dimensões: GameServerGroup, InstanceType
InstanceInterruptions	Instâncias spot que foram interrompidas devido à disponibilidade limitada. Unidades: contagem CloudWatch Estatísticas relevantes da Amazon: soma Dimensões: GameServerGroup, InstanceType

Registro em log Amazon GameLift Servers Chamadas de API com AWS CloudTrail

Amazon GameLift Servers é integrado com AWS CloudTrail, um serviço que fornece um registro das ações realizadas por um usuário, função ou AWS serviço no Amazon GameLift Servers. CloudTrail captura todas as chamadas de API para Amazon GameLift Servers como eventos. As chamadas capturadas incluem chamadas do Amazon GameLift Servers chamadas de console e código para o Amazon GameLift Servers Operações de API. Se você criar uma trilha, poderá habilitar a entrega contínua de CloudTrail eventos para um bucket do Amazon S3, incluindo eventos para Amazon GameLift Servers. Se você não configurar uma trilha, ainda poderá ver os eventos mais recentes no CloudTrail console no Histórico de eventos. Usando as informações coletadas por CloudTrail, você pode determinar a solicitação que foi feita para Amazon GameLift Servers, o endereço IP a partir do qual a solicitação foi feita, quem fez a solicitação, quando ela foi feita e detalhes adicionais.

Para saber mais sobre isso CloudTrail, consulte o [Guia AWS CloudTrail do usuário](#).

Amazon GameLift Servers informações em CloudTrail

CloudTrail é ativado no seu Conta da AWS quando você cria a conta. Quando a atividade ocorre em Amazon GameLift Servers, essa atividade é registrada em um CloudTrail evento junto com outros eventos AWS de serviço no histórico de eventos. Você pode visualizar, pesquisar e baixar eventos recentes no seu Conta da AWS. Para obter mais informações, consulte [Visualização de eventos com histórico de CloudTrail eventos](#).

Para um registro contínuo de eventos em seu Conta da AWS, incluindo eventos para Amazon GameLift Servers, crie uma trilha. Uma trilha permite CloudTrail entregar arquivos de log para um bucket do Amazon S3. Por padrão, quando você cria uma trilha no console, ela é aplicada a todas as Regiões da AWS. A trilha registra eventos de todas as regiões na AWS partição e entrega os arquivos de log ao bucket do Amazon S3 que você especificar. Além disso, você pode configurar outros AWS serviços para analisar e agir com base nos dados de eventos coletados nos CloudTrail registros. Para obter mais informações, consulte:

- [Visão geral da criação de uma trilha](#)
- [CloudTrail serviços e integrações suportados](#)
- [Configurando notificações do Amazon SNS para CloudTrail](#)
- [Recebendo arquivos de CloudTrail log de várias regiões](#) e [Recebendo arquivos de CloudTrail log de várias contas](#)

Todos Amazon GameLift Servers as ações são registradas CloudTrail e documentadas no [Amazon GameLift Servers Referência da API](#) . Por exemplo, chamadas `CreatePlayerSession` e `UpdateGameSession` ações geram entradas nos arquivos de CloudTrail log. `CreateGameSession`

Cada entrada de log ou evento contém informações sobre quem gerou a solicitação. As informações de identidade ajudam a determinar o seguinte:

- Se a solicitação foi feita com credenciais de usuário root ou AWS Identity and Access Management (IAM).
- Se a solicitação foi feita com credenciais de segurança temporárias de uma função ou de um usuário federado.
- Se a solicitação foi feita por outro AWS serviço.

Para obter mais informações, consulte [Elemento `userIdentity` do CloudTrail](#) .

Noções básicas Amazon GameLift Servers entradas do arquivo de log

Uma trilha é uma configuração que permite a entrega de eventos como arquivos de log para um bucket do Amazon S3 que você especificar. CloudTrail os arquivos de log contêm uma ou mais entradas de log. Um evento representa uma única solicitação de qualquer fonte e inclui informações sobre a ação solicitada, a data e a hora da ação, os parâmetros da solicitação e assim por diante. CloudTrail os arquivos de log não são um rastreamento de pilha ordenado das chamadas públicas de API, portanto, eles não aparecem em nenhuma ordem específica.

O exemplo a seguir mostra uma entrada de CloudTrail registro que demonstra as `DescribeFleetAttributes` ações `CreateFleet` e.

```
{
  "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"
  },
]
}

```

Registrando mensagens do servidor em Amazon GameLift Servers

Você pode capturar mensagens personalizadas de seus Amazon GameLift Servers servidores em arquivos de log. A forma como você configura o registro depende do uso de servidores personalizados ou Amazon GameLift Servers Realtime (consulte as subseções apropriadas neste capítulo).

Tópicos

- [Registro em log de mensagens do servidor \(servidores personalizados\)](#)

- [Registrando mensagens do servidor \(Amazon GameLift ServersRealtime\)](#)

Registro em log de mensagens do servidor (servidores personalizados)

Você pode capturar mensagens personalizadas do servidor de seus servidores Amazon GameLift Servers personalizados em arquivos de log. Para saber mais sobre como registrar Amazon GameLift ServersRealtime, consulte [Registrando mensagens do servidor \(Amazon GameLift ServersRealtime\)](#).

Important

Há um limite no tamanho de um arquivo de log por sessão de jogo (veja [Amazon GameLift Serversendpoints e cotas](#) no Referência geral da AWS). Quando uma sessão de jogo termina, Amazon GameLift Servers carrega os registros do servidor para o Amazon Simple Storage Service (Amazon S3). Amazon GameLift Servers não carregará registros que excedam o limite. Os troncos podem crescer muito rapidamente e exceder o limite de tamanho. Você deve monitorar seus logs e limitar a saída do log somente às mensagens necessárias.

Configurando o registro em log para servidores personalizados

Com servidores Amazon GameLift Servers personalizados, você escreve seu próprio código para realizar o registro, que você configura como parte da configuração do processo do servidor. Amazon GameLift Servers usa sua configuração de registro para identificar os arquivos que devem ser enviados para o Amazon S3 no final de cada sessão de jogo.

As instruções a seguir mostram como configurar o registro usando exemplos de código simplificados:

C++

Para configurar o registro em log (C++)

1. Crie um vetor de strings que sejam caminhos de diretório para arquivos de log do servidor de jogos.

```
std::string serverLog("serverOut.log");           // Example server log file
std::vector<std::string> logPaths;
logPaths.push_back(serverLog);
```


2. Forneça seu vetor como o [LogParameters](#) do seu [ProcessParameters](#) objeto.

```
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. Forneça o [ProcessParameters](#) objeto ao chamar [ProcessReady\(\)](#).

```
Aws::GameLift::GenericOutcome outcome =
    Aws::GameLift::Server::ProcessReady(processReadyParameter);
```

Para um exemplo mais completo, consulte [ProcessReady\(\)](#).

C#

Para configurar o registro em log (C#)

1. Crie uma lista de strings que são caminhos de diretório para arquivos de log do servidor de jogos.

```
List<string> logPaths = new List<string>();
logPaths.Add("C:\\game\\serverOut.txt");    // Example of a log file that the
game server writes
```

2. Forneça sua lista como a [LogParameters](#) do seu [ProcessParameters](#) objeto.

```
var processReadyParameter = new ProcessParameters(
    this.OnGameSession,
    this.OnProcessTerminate,
    this.OnHealthCheck,
    this.OnGameSessionUpdate,
    port,
    new LogParameters(logPaths));
```

3. Forneça o [ProcessParameters](#) objeto ao chamar [ProcessReady\(\)](#).

```
var processReadyOutcome =
```

```
GameLiftServerAPI.ProcessReady(processReadyParameter);
```

Para um exemplo mais completo, consulte [ProcessReady\(\)](#).

Gravar em registros

Seus arquivos de log existem após o início do processo do servidor. Você poderá gravar nos registros usando qualquer método para gravar nos arquivos. Para capturar toda a saída padrão e a saída de erro do seu servidor, remapeie os fluxos de saída para arquivos de log, como nos exemplos a seguir:

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"));
```

Acessando logs do servidor

Quando uma sessão de jogo termina, armazena Amazon GameLift Servers automaticamente os registros em um bucket do Amazon S3 e os retém por 14 dias. Para obter a localização dos registros de uma sessão de jogo, você pode usar a operação da [GetGameSessionLogUrlAPI](#). Para baixar os registros, use a URL que a operação retorna.

Registrando mensagens do servidor (Amazon GameLift ServersRealtime)

Você pode capturar mensagens personalizadas do servidor a partir dos seus Amazon GameLift Servers Realtime arquivos de log. Para saber mais sobre o registro em servidores personalizados, consulte [Registro em log de mensagens do servidor \(servidores personalizados\)](#).

Há diferentes tipos de mensagens que você pode enviar para seus arquivos de log (consulte [Registrar mensagens no script do seu servidor](#)). Além das mensagens personalizadas, as mensagens do sistema Amazon GameLift Servers Realtime de saída usam os mesmos tipos de

mensagens e gravam nos mesmos arquivos de log. Você pode ajustar o nível de registro da sua frota para reduzir a quantidade de mensagens de registro que seus servidores geram (consulte [Ajustar logs](#)).

Important

Há um limite no tamanho de um arquivo de log por sessão de jogo (veja [Amazon GameLift Servers endpoints e cotas](#) no Referência geral da AWS). Quando uma sessão de jogo termina, Amazon GameLift Servers carrega os registros do servidor para o Amazon Simple Storage Service (Amazon S3). Amazon GameLift Servers não carregará registros que excedam o limite. Os troncos podem crescer muito rapidamente e exceder o limite de tamanho. Você deverá monitorar seus registros e limitar a saída do log somente às mensagens necessárias.

Registrar mensagens no script do seu servidor

Você pode gerar mensagens personalizadas no [script do seu Amazon GameLift Servers Realtime](#). Use as etapas a seguir para enviar mensagens do servidor para um arquivo de log:

1. Crie uma variável para manter a referência ao objeto logger.

```
var logger;
```

2. Na função de `init()`, obtenha o registrador do objeto de sessão e atribua-o à sua variável do registrador.

```
function init(rtSession) {  
    session = rtSession;  
    logger = session.getLogger();  
}
```

3. Chame a função apropriada no registrador para gerar uma mensagem.

Mensagens de depuração

```
logger.debug("This is my debug message...");
```

Mensagens informativas

```
logger.info("This is my info message...");
```

Mensagens de aviso

```
logger.warn("This is my warn message...");
```

Mensagens de erro

```
logger.error("This is my error message...");
```

Mensagens de erro fatais

```
logger.fatal("This is my fatal error message...");
```

Mensagens de erro fatais da experiência do cliente

```
logger.cxfatal("This is my customer experience fatal error message...");
```

Para obter um exemplo das instruções de registro em um script, consulte [Amazon GameLift ServersRealtimeexemplo de script](#).

A saída nos arquivos de log indica o tipo de mensagem (DEBUG,INFO,WARN,ERROR,FATAL,CXFATAL), conforme mostrado nas seguintes linhas de um exemplo de registro:

```
09 Sep 2021 11:46:32,970 [INFO] (gamelift.js) 215: Calling GameLiftServerAPI.InitSDK...
09 Sep 2021 11:46:32,993 [INFO] (gamelift.js) 220: GameLiftServerAPI.InitSDK succeeded
09 Sep 2021 11:46:32,993 [INFO] (gamelift.js) 223: Waiting for Realtime server to
start...
09 Sep 2021 11:46:33,15 [WARN] (index.js) 204: Connection is INSECURE. Messages will be
sent/received as plaintext.
```

Acessando logs do servidor

Quando uma sessão de jogo termina, armazena Amazon GameLift Servers automaticamente os registros no Amazon S3 e os retém por 14 dias. Você pode usar a [chamada de](#)

[GetGameSessionLogUrl API](#) para obter a localização dos registros de uma sessão de jogo. Use o URL retornado pela chamada de API para baixar os registros.

Ajustar logs

Os troncos podem crescer muito rapidamente e exceder o limite de tamanho. Você deverá monitorar seus registros e limitar a saída do log somente às mensagens necessárias. Pois Amazon GameLift ServersRealtime, você pode ajustar o nível de registro fornecendo um parâmetro na configuração de tempo de execução da sua frota no formulário `loggingLevel: LOGGING_LEVEL`, onde **LOGGING_LEVEL** está um dos seguintes valores:

1. debug
2. info (padrão)
3. warn
4. error
5. fatal
6. cxfatal

Essa lista está ordenada da menos severa (debug) à mais grave (cxfatal). Você define um único `loggingLevel` e o servidor só registrará mensagens nesse nível de gravidade ou em um nível de gravidade mais alto. Por exemplo, a configuração `loggingLevel:error` fará com que todos os servidores da sua frota apenas gravem `error`, `fatal`, e `cxfatal` mensagens ao log.

É possível definir o nível de registro da frota ao criá-lo ou depois que ele já existe. Alterar o nível de registro da sua frota após sua execução afetará apenas os registros das sessões de jogo criadas após a atualização. Os registros de qualquer sessão de jogo existente não serão afetados. Se você não definir um nível de registro ao criar sua frota, seus servidores definirão o nível de registro como padrão `info`. Consulte as seções a seguir para obter instruções sobre como definir o nível de registro.

Definir o nível de registro ao criar uma Amazon GameLift ServersRealtime frota (console)

Siga as instruções em [Crie uma EC2 frota Amazon GameLift Servers gerenciada](#) para criar sua frota, com a seguinte adição:

- Na subetapa de Alocação de processos do servidor da etapa de Gerenciamento de processos, forneça o par de valores-chave do nível de registro (como `loggingLevel:error`) como um valor para os Parâmetros de inicialização. Use um caractere não alfanumérico (exceto

vírgula) para separar o nível de registro de quaisquer parâmetros adicionais (por exemplo, `loggingLevel:error +map Winter444`).

Definindo o nível de registro ao criar uma Amazon GameLift ServersRealtime frota (AWS CLI)

Siga as instruções em [Crie uma EC2 frota Amazon GameLift Servers gerenciada](#) para criar sua frota, com a seguinte adição:

- No argumento do parâmetro `--runtime-configuration` para [create-fleet](#), forneça o par de valores-chave do nível de registro (como `loggingLevel:error`) como um valor para `Parameters`. Use um caractere não alfanumérico (exceto vírgula) para separar o nível de registro de quaisquer parâmetros adicionais. Veja o exemplo a seguir:

```
--runtime-configuration "GameSessionActivationTimeoutSeconds=60,
                          MaxConcurrentGameSessionActivations=2,
                          ServerProcesses=[{LaunchPath=/local/game/myRealtimeLaunchScript.js,
                          Parameters=loggingLevel:error +map Winter444,
                          ConcurrentExecutions=10}]"
```

Definindo o nível de registro para uma Amazon GameLift ServersRealtime frota em execução (console)

Siga as instruções em [Atualizar uma configuração de Amazon GameLift Servers frota](#) para atualizar sua frota usando o Amazon GameLift Servers console, com a seguinte adição:

- Na página Editar frota, em Alocação de processos do servidor, forneça o par de valores-chave do nível de registro (como `loggingLevel:error`) como um valor para os Parâmetros de inicialização. Use um caractere não alfanumérico (exceto vírgula) para separar o nível de registro de quaisquer parâmetros adicionais (por exemplo, `loggingLevel:error +map Winter444`).

Definindo o nível de registro para uma Amazon GameLift ServersRealtime frota em execução (AWS CLI)

Siga as instruções em [Atualizar uma configuração de Amazon GameLift Servers frota](#) para atualizar sua frota usando o AWS CLI, com a seguinte adição:

- No argumento do parâmetro `--runtime-configuration` para [update-runtime-configuration](#), forneça o par de valores-chave do nível de registro (como

`loggingLevel:error`) como um valor para `Parameters`. Use um caractere não alfanumérico (exceto vírgula) para separar o nível de registro de quaisquer parâmetros adicionais. Veja o exemplo a seguir:

```
--runtime-configuration "GameSessionActivationTimeoutSeconds=60,  
    MaxConcurrentGameSessionActivations=2,  
    ServerProcesses=[{LaunchPath=/local/game/myRealtimeLaunchScript.js,  
        Parameters=loggingLevel:error +map Winter444,  
        ConcurrentExecutions=10}]"
```

Segurança em Amazon GameLift Servers

Se você estiver usando Amazon GameLift Servers FleetIQ como um recurso independente da Amazon EC2, consulte [Segurança na Amazon EC2 no](#) Guia do EC2 usuário da Amazon.

A segurança na nuvem AWS é a maior prioridade. Como cliente da AWS, você se beneficiará de data centers e arquiteturas de rede criados para atender aos requisitos das empresas com as maiores exigências de segurança.

A segurança é uma responsabilidade compartilhada entre você AWS e você. O [modelo de responsabilidade compartilhada](#) descreve isso como segurança da nuvem e segurança na nuvem:

- **Segurança da nuvem** — AWS é responsável por proteger a infraestrutura que executa AWS os serviços na AWS nuvem. AWS também fornece serviços que você pode usar com segurança. Auditores terceirizados testam e verificam regularmente a eficácia de nossa segurança como parte dos programas de [AWS conformidade dos programas](#) de de . Para saber mais sobre os programas de conformidade que se aplicam ao Amazon GameLift Servers, veja [AWS serviços no escopo por programa de conformidade AWS](#) .
- **Segurança na nuvem** — Sua responsabilidade é determinada pelo AWS serviço que você usa. Você também é responsável por outros fatores, incluindo a confidencialidade de seus dados, os requisitos da sua empresa e os AWS regulamentos aplicáveis.

Esta documentação ajuda você a entender como aplicar o modelo de responsabilidade compartilhada ao usar Amazon GameLift Servers. Os tópicos a seguir mostram como configurar Amazon GameLift Servers para atender aos seus objetivos de segurança e conformidade. Você também aprende a usar outros AWS serviços que ajudam você a monitorar e proteger seu Amazon GameLift Servers recursos.

Tópicos

- [Proteção de dados no Amazon GameLift Servers](#)
- [Gerenciamento de identidade e acesso para Amazon GameLift Servers](#)
- [Registro e monitoramento com Amazon GameLift Servers](#)
- [Validação de conformidade para Amazon GameLift Servers](#)
- [Resiliência em Amazon GameLift Servers](#)
- [Segurança da infraestrutura em Amazon GameLift Servers](#)

- [Análise de configuração e vulnerabilidade em Amazon GameLift Servers](#)
- [Melhores práticas de segurança do Amazon GameLift Servers](#)

Proteção de dados no Amazon GameLift Servers

Se você estiver usando Amazon GameLift Servers FleetIQ como um recurso independente com a Amazon EC2, consulte [Segurança na Amazon EC2 no](#) Guia do EC2 usuário da Amazon.

O modelo de [responsabilidade AWS compartilhada modelo](#) se aplica à proteção de dados em Amazon GameLift Servers. Conforme descrito neste modelo, AWS é responsável por proteger a infraestrutura global que executa todos os Nuvem AWS. Você é responsável por manter o controle sobre o conteúdo hospedado nessa infraestrutura. Você também é responsável pelas tarefas de configuração e gerenciamento de segurança dos Serviços da AWS que usa. Para obter mais informações sobre a privacidade de dados, consulte as [Data Privacy FAQ](#). Para obter mais informações sobre a proteção de dados na Europa, consulte a postagem do blog [AWS Shared Responsibility Model and RGPD](#) no Blog de segurança da AWS .

Para fins de proteção de dados, recomendamos que você proteja Conta da AWS as credenciais e configure usuários individuais com AWS IAM Identity Center ou AWS Identity and Access Management (IAM). Dessa maneira, cada usuário receberá apenas as permissões necessárias para cumprir suas obrigações de trabalho. Recomendamos também que você proteja seus dados das seguintes formas:

- Use uma autenticação multifator (MFA) com cada conta.
- Use SSL/TLS para se comunicar com AWS os recursos. Exigimos TLS 1.2 e recomendamos TLS 1.3.
- Configure a API e o registro de atividades do usuário com AWS CloudTrail. Para obter informações sobre o uso de CloudTrail trilhas para capturar AWS atividades, consulte Como [trabalhar com CloudTrail trilhas](#) no Guia AWS CloudTrail do usuário.
- Use soluções de AWS criptografia, juntamente com todos os controles de segurança padrão Serviços da AWS.
- Use serviços gerenciados de segurança avançada, como o Amazon Macie, que ajuda a localizar e proteger dados sigilosos armazenados no Amazon S3.
- Se você precisar de módulos criptográficos validados pelo FIPS 140-3 ao acessar AWS por meio de uma interface de linha de comando ou de uma API, use um endpoint FIPS. Para obter

mais informações sobre os endpoints FIPS disponíveis, consulte [Federal Information Processing Standard \(FIPS\) 140-3](#).

É altamente recomendável que nunca sejam colocadas informações confidenciais ou sigilosas, como endereços de e-mail de clientes, em tags ou campos de formato livre, como um campo Nome. Isso inclui quando você trabalha com Amazon GameLift Servers ou Serviços da AWS usa o console, a API ou AWS SDKs. AWS CLI Quaisquer dados inseridos em tags ou em campos de texto de formato livre usados para nomes podem ser usados para logs de faturamento ou de diagnóstico. Se você fornecer um URL para um servidor externo, é fortemente recomendável que não sejam incluídas informações de credenciais no URL para validar a solicitação nesse servidor.

Os dados específicos do Amazon GameLift Servers são tratados da seguinte forma:

- As compilações do servidor de jogos e os scripts para Amazon GameLift Servers os quais você carrega são armazenados no Amazon S3. Não há acesso direto do cliente a esses dados após fazer upload. Um usuário autorizado pode obter acesso temporário para fazer upload de arquivos, mas não pode visualizar nem atualizar os arquivos no Amazon S3 diretamente. Para excluir scripts e compilações, use o console do Amazon GameLift Servers ou a API de serviço.
- Os dados de log da sessão de jogo são armazenados no Amazon S3 durante um período limitado de tempo após a conclusão da sessão de jogo. Os usuários autorizados podem acessar os dados de log fazendo download deles por meio de um link no console do Amazon GameLift Servers ou por chamadas para a API de serviço.
- Os dados de métricas e eventos são armazenados no Amazon GameLift Servers e podem ser acessados pelo console do Amazon GameLift Servers ou por chamadas para a API de serviço. É possível recuperar os dados de frotas, instâncias, colocações em sessões de jogos, tíquetes de marcação de jogos, sessões de jogos e sessões de jogadores. Os dados também podem ser acessados por meio da Amazon CloudWatch e da CloudWatch Events.
- Os dados fornecidos pelo cliente são armazenados no Amazon GameLift Servers. Os usuários autorizados podem acessá-los por chamadas para a API de serviço. Os dados potencialmente confidenciais podem incluir dados de jogadores, dados da sessão de jogador e da sessão de jogo (incluindo informações de conexão), dados da marcação de jogos e assim por diante.

 Note

Se você fornecer um jogador personalizado IDs em suas solicitações, espera-se que esses valores sejam anônimos UUIDs e não contenham informações de identificação do jogador.

Para obter mais informações sobre proteção de dados, consulte a publicação [AWS modelo de responsabilidade compartilhada e GDPR](#) no blog de segurança da AWS .

Criptografia inativa

A criptografia em repouso de dados específicos do Amazon GameLift Servers é tratada da seguinte forma:

- As compilações e scripts do servidor de jogos são armazenados em buckets do Amazon S3 com criptografia no lado do servidor.
- Os dados fornecidos pelo cliente são armazenados no Amazon GameLift Servers em um formato criptografado.

Criptografia em trânsito

As conexões com o Amazon GameLift Servers APIs são feitas por meio de uma conexão segura (SSL) e autenticadas usando o [AWS Signature versão 4](#) (ao se conectar por meio da AWS CLI ou do AWS SDK, a assinatura é feita automaticamente). A autenticação é gerenciada usando as políticas de acesso definidas pelo IAM para as credenciais de segurança usadas para fazer a conexão.

A comunicação direta entre clientes de jogos e servidores de jogos é a seguinte:

- Para servidores de jogos personalizados hospedados em recursos do Amazon GameLift Servers, a comunicação não envolve o serviço Amazon GameLift Servers. A criptografia dessa comunicação é de responsabilidade do cliente. É possível usar frotas habilitadas para TLS para que os clientes do jogo autenticuem o servidor do jogo na conexão e criptografem toda a comunicação entre o cliente do jogo e o servidor do jogo.
- Pois Amazon GameLift Servers Realtime com a geração de certificados TLS ativada, o tráfego entre o cliente do jogo e Realtime os servidores usando o SDK do cliente Realtime é criptografado em trânsito. O tráfego TCP é criptografado usando TLS 1.2 e o tráfego UDP é criptografado usando DTLS 1.2.

Privacidade do tráfego entre redes

É possível acessar remotamente as instâncias do Amazon GameLift Servers com segurança. Para instâncias que usam Linux, o SSH fornece um canal de comunicação seguro para acesso remoto. Para instâncias que estão executando o Windows, use um cliente de protocolo de RDP (área de

trabalho remota). Com Amazon GameLift ServersFleetIQ, o acesso remoto às suas instâncias usando o AWS Systems Manager Session Manager e o Run Command é criptografado usando TLS 1.2, e as solicitações para criar uma conexão são assinadas usando SigV4. Para obter ajuda sobre como se conectar a uma instância gerenciada do Amazon GameLift Servers, consulte [Conecte-se remotamente a Amazon GameLift Servers instâncias de frota](#).

Gerenciamento de identidade e acesso para Amazon GameLift Servers

AWS Identity and Access Management (IAM) é uma ferramenta AWS service (Serviço da AWS) que ajuda o administrador a controlar com segurança o acesso aos AWS recursos. Os administradores do IAM controlam quem pode ser autenticado (conectado) e autorizado (tem permissões) para usar Amazon GameLift Servers recursos. O IAM é um AWS service (Serviço da AWS) que você pode usar sem custo adicional.

Tópicos

- [Público](#)
- [Autenticar com identidades](#)
- [Gerenciar o acesso usando políticas](#)
- [Como Amazon GameLift Servers funciona com o IAM](#)
- [Exemplos de políticas baseadas em identidade para Amazon GameLift Servers](#)
- [Solução de problemas Amazon GameLift Servers identidade e acesso](#)
- [AWS políticas gerenciadas para Amazon GameLift Servers](#)

Público

A forma como você usa AWS Identity and Access Management (IAM) difere, dependendo do trabalho que você faz no Amazon GameLift Servers.

Usuário do serviço — Se você usar o Amazon GameLift Servers serviço para fazer seu trabalho, então seu administrador fornece as credenciais e permissões de que você precisa. À medida que você usa mais Amazon GameLift Servers recursos para fazer seu trabalho, talvez você precise de permissões adicionais. Compreenda como o acesso é gerenciado pode ajudar a solicitar as permissões corretas ao administrador. Se você não conseguir acessar um recurso no Amazon GameLift Servers, consulte [Solução de problemas Amazon GameLift Servers identidade e acesso](#).

Administrador de serviços — Se você é responsável por Amazon GameLift Servers recursos em sua empresa, você provavelmente tem acesso total aos Amazon GameLift Servers. É seu trabalho determinar quais Amazon GameLift Servers recursos e recursos que seus usuários do serviço devem acessar. Envie as solicitações ao administrador do IAM para alterar as permissões dos usuários de serviço. Revise as informações nesta página para compreender os conceitos básicos do IAM. Para saber mais sobre como sua empresa pode usar o IAM com Amazon GameLift Servers, consulte [Como Amazon GameLift Servers funciona com o IAM](#).

Administrador do IAM — Se você for um administrador do IAM, talvez queira saber detalhes sobre como criar políticas para gerenciar o acesso ao Amazon GameLift Servers. Para ver um exemplo Amazon GameLift Servers políticas baseadas em identidade que você pode usar no IAM, consulte [Exemplos de políticas baseadas em identidade para Amazon GameLift Servers](#)

Autenticar com identidades

A autenticação é a forma como você faz login AWS usando suas credenciais de identidade. Você deve estar autenticado (conectado AWS) como o Usuário raiz da conta da AWS, como usuário do IAM ou assumindo uma função do IAM.

Você pode entrar AWS como uma identidade federada usando credenciais fornecidas por meio de uma fonte de identidade. AWS IAM Identity Center Usuários (IAM Identity Center), a autenticação de login único da sua empresa e suas credenciais do Google ou do Facebook são exemplos de identidades federadas. Quando você faz login como identidade federada, o administrador já configurou anteriormente a federação de identidades usando perfis do IAM. Ao acessar AWS usando a federação, você está assumindo indiretamente uma função.

Dependendo do tipo de usuário que você é, você pode entrar no AWS Management Console ou no portal de AWS acesso. Para obter mais informações sobre como fazer login em AWS, consulte [Como fazer login Conta da AWS](#) no Guia do Início de Sessão da AWS usuário.

Se você acessar AWS programaticamente, AWS fornece um kit de desenvolvimento de software (SDK) e uma interface de linha de comando (CLI) para assinar criptograficamente suas solicitações usando suas credenciais. Se você não usa AWS ferramentas, você mesmo deve assinar as solicitações. Para obter mais informações sobre como usar o método recomendado para designar solicitações por conta própria, consulte [Versão 4 do AWS Signature para solicitações de API](#) no Guia do usuário do IAM.

Independente do método de autenticação usado, também pode ser necessário fornecer informações adicionais de segurança. Por exemplo, AWS recomenda que você use a autenticação multifator

(MFA) para aumentar a segurança da sua conta. Para saber mais, consulte [Autenticação multifator](#) no Guia do usuário do AWS IAM Identity Center e [Usar a autenticação multifator da AWS no IAM](#) no Guia do usuário do IAM.

Conta da AWS usuário root

Ao criar uma Conta da AWS, você começa com uma identidade de login que tem acesso completo a todos Serviços da AWS os recursos da conta. Essa identidade é chamada de usuário Conta da AWS raiz e é acessada fazendo login com o endereço de e-mail e a senha que você usou para criar a conta. É altamente recomendável não usar o usuário-raiz para tarefas diárias. Proteja as credenciais do usuário-raiz e use-as para executar as tarefas que somente ele puder executar. Para obter a lista completa das tarefas que exigem login como usuário-raiz, consulte [Tarefas que exigem credenciais de usuário-raiz](#) no Guia do Usuário do IAM.

Identidade federada

Como prática recomendada, exija que usuários humanos, incluindo usuários que precisam de acesso de administrador, usem a federação com um provedor de identidade para acessar Serviços da AWS usando credenciais temporárias.

Uma identidade federada é um usuário do seu diretório de usuários corporativo, de um provedor de identidade da web AWS Directory Service, do diretório do Identity Center ou de qualquer usuário que acesse usando credenciais fornecidas Serviços da AWS por meio de uma fonte de identidade. Quando as identidades federadas são acessadas Contas da AWS, elas assumem funções, e as funções fornecem credenciais temporárias.

Para o gerenciamento de acesso centralizado, é recomendável usar o AWS IAM Identity Center. Você pode criar usuários e grupos no IAM Identity Center ou pode se conectar e sincronizar com um conjunto de usuários e grupos em sua própria fonte de identidade para uso em todos os seus Contas da AWS aplicativos. Para obter mais informações sobre o Centro de Identidade do IAM, consulte [O que é o Centro de Identidade do IAM?](#) no Guia do Usuário do AWS IAM Identity Center .

Usuários e grupos do IAM

Um [usuário do IAM](#) é uma identidade dentro da sua Conta da AWS que tem permissões específicas para uma única pessoa ou aplicativo. Sempre que possível, é recomendável contar com credenciais temporárias em vez de criar usuários do IAM com credenciais de longo prazo, como senhas e chaves de acesso. No entanto, se você tiver casos de uso específicos que exijam credenciais de longo prazo com usuários do IAM, é recomendável alternar as chaves de acesso. Para obter mais

informações, consulte [Alternar as chaves de acesso regularmente para casos de uso que exijam credenciais de longo prazo](#) no Guia do Usuário do IAM.

Um [grupo do IAM](#) é uma identidade que especifica uma coleção de usuários do IAM. Não é possível fazer login como um grupo. É possível usar grupos para especificar permissões para vários usuários de uma vez. Os grupos facilitam o gerenciamento de permissões para grandes conjuntos de usuários. Por exemplo, você pode ter um grupo chamado IAMAdminse conceder a esse grupo permissões para administrar recursos do IAM.

Usuários são diferentes de perfis. Um usuário é exclusivamente associado a uma pessoa ou a uma aplicação, mas um perfil pode ser assumido por qualquer pessoa que precisar dele. Os usuários têm credenciais permanentes de longo prazo, mas os perfis fornecem credenciais temporárias. Para saber mais, consulte [Casos de uso para usuários do IAM](#) no Guia do usuário do IAM.

Perfis do IAM

Uma [função do IAM](#) é uma identidade dentro da sua Conta da AWS que tem permissões específicas. Ele é semelhante a um usuário do IAM, mas não está associado a uma pessoa específica. Para assumir temporariamente uma função do IAM no AWS Management Console, você pode [alternar de um usuário para uma função do IAM \(console\)](#). Você pode assumir uma função chamando uma operação de AWS API AWS CLI ou usando uma URL personalizada. Para obter mais informações sobre métodos para usar perfis, consulte [Métodos para assumir um perfil](#) no Guia do usuário do IAM.

Perfis do IAM com credenciais temporárias são úteis nas seguintes situações:

- **Acesso de usuário federado:** para atribuir permissões a identidades federadas, é possível criar um perfil e definir permissões para ele. Quando uma identidade federada é autenticada, essa identidade é associada ao perfil e recebe as permissões definidas por ele. Para ter mais informações sobre perfis para federação, consulte [Criar um perfil para um provedor de identidade de terceiros \(federação\)](#) no Guia do usuário do IAM. Se usar o Centro de Identidade do IAM, configure um conjunto de permissões. Para controlar o que suas identidades podem acessar após a autenticação, o Centro de Identidade do IAM correlaciona o conjunto de permissões a um perfil no IAM. Para obter informações sobre conjuntos de permissões, consulte [Conjuntos de Permissões](#) no Guia do Usuário do AWS IAM Identity Center .
- **Permissões temporárias para usuários do IAM:** um usuário ou um perfil do IAM pode presumir um perfil do IAM para obter temporariamente permissões diferentes para uma tarefa específica.
- **Acesso entre contas:** é possível usar um perfil do IAM para permitir que alguém (uma entidade principal confiável) em outra conta acesse recursos em sua conta. Os perfis são a principal forma de conceder acesso entre contas. No entanto, com alguns Serviços da AWS, você pode anexar

uma política diretamente a um recurso (em vez de usar uma função como proxy). Para conhecer a diferença entre perfis e políticas baseadas em recurso para acesso entre contas, consulte [Acesso a recursos entre contas no IAM](#) no Guia do usuário do IAM.

- **Acesso entre serviços** — Alguns Serviços da AWS usam recursos em outros Serviços da AWS. Por exemplo, quando você faz uma chamada em um serviço, é comum que esse serviço execute aplicativos na Amazon EC2 ou armazene objetos no Amazon S3. Um serviço pode fazer isso usando as permissões da entidade principal da chamada, usando um perfil de serviço ou um perfil vinculado ao serviço.
- **Sessões de acesso direto (FAS)** — Quando você usa um usuário ou uma função do IAM para realizar ações AWS, você é considerado principal. Ao usar alguns serviços, você pode executar uma ação que inicia outra ação em um serviço diferente. O FAS usa as permissões do diretor chamando um AWS service (Serviço da AWS), combinadas com a solicitação AWS service (Serviço da AWS) para fazer solicitações aos serviços posteriores. As solicitações do FAS são feitas somente quando um serviço recebe uma solicitação que requer interações com outros Serviços da AWS ou com recursos para ser concluída. Nesse caso, você precisa ter permissões para executar ambas as ações. Para obter detalhes da política ao fazer solicitações de FAS, consulte [Sessões de acesso direto](#).
- **Perfil de serviço**: um perfil de serviço é um [perfil do IAM](#) que um serviço assume para executar ações em seu nome. Um administrador do IAM pode criar, modificar e excluir um perfil de serviço do IAM. Para obter mais informações, consulte [Criar um perfil para delegar permissões a um AWS service \(Serviço da AWS\)](#) no Guia do Usuário do IAM.
- **Função vinculada ao serviço** — Uma função vinculada ao serviço é um tipo de função de serviço vinculada a um AWS service (Serviço da AWS). O serviço pode presumir o perfil de executar uma ação em seu nome. As funções vinculadas ao serviço aparecem em você Conta da AWS e são de propriedade do serviço. Um administrador do IAM pode visualizar, mas não editar as permissões para perfis vinculados ao serviço.
- **Aplicativos em execução na Amazon EC2** — Você pode usar uma função do IAM para gerenciar credenciais temporárias para aplicativos que estão sendo executados em uma EC2 instância e fazendo solicitações AWS CLI de AWS API. Isso é preferível ao armazenamento de chaves de acesso na EC2 instância. Para atribuir uma AWS função a uma EC2 instância e disponibilizá-la para todos os aplicativos, você cria um perfil de instância anexado à instância. Um perfil de instância contém a função e permite que os programas em execução na EC2 instância recebam credenciais temporárias. Para obter mais informações, consulte [Usar uma função do IAM para conceder permissões a aplicativos executados em EC2 instâncias da Amazon](#) no Guia do usuário do IAM.

Gerenciar o acesso usando políticas

Você controla o acesso AWS criando políticas e anexando-as a AWS identidades ou recursos. Uma política é um objeto AWS que, quando associada a uma identidade ou recurso, define suas permissões. AWS avalia essas políticas quando um principal (usuário, usuário raiz ou sessão de função) faz uma solicitação. As permissões nas políticas determinam se a solicitação será permitida ou negada. A maioria das políticas é armazenada AWS como documentos JSON. Para obter mais informações sobre a estrutura e o conteúdo de documentos de políticas JSON, consulte [Visão geral das políticas JSON](#) no Guia do usuário do IAM.

Os administradores podem usar políticas AWS JSON para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos e em que condições.

Por padrão, usuários e perfis não têm permissões. Para conceder permissão aos usuários para executar ações nos recursos que eles precisam, um administrador do IAM pode criar políticas do IAM. O administrador pode então adicionar as políticas do IAM aos perfis e os usuários podem assumir os perfis.

As políticas do IAM definem permissões para uma ação independentemente do método usado para executar a operação. Por exemplo, suponha que você tenha uma política que permite a ação `iam:GetRole`. Um usuário com essa política pode obter informações de função da AWS Management Console AWS CLI, da ou da AWS API.

Políticas baseadas em identidade

As políticas baseadas em identidade são documentos de políticas de permissões JSON que você pode anexar a uma identidade, como usuário, grupo de usuários ou perfil do IAM. Essas políticas controlam quais ações os usuários e perfis podem realizar, em quais recursos e em que condições. Para saber como criar uma política baseada em identidade, consulte [Definir permissões personalizadas do IAM com as políticas gerenciadas pelo cliente](#) no Guia do Usuário do IAM.

As políticas baseadas em identidade podem ser categorizadas como políticas em linha ou políticas gerenciadas. As políticas em linha são anexadas diretamente a um único usuário, grupo ou perfil. As políticas gerenciadas são políticas autônomas que você pode associar a vários usuários, grupos e funções em seu Conta da AWS. As políticas AWS gerenciadas incluem políticas gerenciadas e políticas gerenciadas pelo cliente. Para saber como escolher entre uma política gerenciada ou uma política em linha, consulte [Escolher entre políticas gerenciadas e políticas em linha](#) no Guia do usuário do IAM.

Políticas baseadas em recursos

Políticas baseadas em recursos são documentos de políticas JSON que você anexa a um recurso. São exemplos de políticas baseadas em recursos as políticas de confiança de perfil do IAM e as políticas de bucket do Amazon S3. Em serviços compatíveis com políticas baseadas em recursos, os administradores de serviço podem usá-las para controlar o acesso a um recurso específico. Para o atributo ao qual a política está anexada, a política define quais ações uma entidade principal especificado pode executar nesse atributo e em que condições. Você deve [especificar uma entidade principal](#) em uma política baseada em recursos. Os diretores podem incluir contas, usuários, funções, usuários federados ou. Serviços da AWS

Políticas baseadas em recursos são políticas em linha localizadas nesse serviço. Você não pode usar políticas AWS gerenciadas do IAM em uma política baseada em recursos.

Listas de controle de acesso (ACLs)

As listas de controle de acesso (ACLs) controlam quais diretores (membros da conta, usuários ou funções) têm permissões para acessar um recurso. ACLs são semelhantes às políticas baseadas em recursos, embora não usem o formato de documento de política JSON.

O Amazon S3 e o AWS WAF Amazon VPC são exemplos de serviços que oferecem suporte. ACLs Para saber mais ACLs, consulte a [visão geral da lista de controle de acesso \(ACL\)](#) no Guia do desenvolvedor do Amazon Simple Storage Service.

Outros tipos de política

AWS oferece suporte a tipos de políticas adicionais menos comuns. Esses tipos de política podem definir o máximo de permissões concedidas a você pelos tipos de política mais comuns.

- Limites de permissões: um limite de permissões é um recurso avançado no qual você define o máximo de permissões que uma política baseada em identidade pode conceder a uma entidade do IAM (usuário ou perfil do IAM). É possível definir um limite de permissões para uma entidade. As permissões resultantes são a interseção das políticas baseadas em identidade de uma entidade com seus limites de permissões. As políticas baseadas em recurso que especificam o usuário ou o perfil no campo `Principal` não são limitadas pelo limite de permissões. Uma negação explícita em qualquer uma dessas políticas substitui a permissão. Para obter mais informações sobre limites de permissões, consulte [Limites de permissões para identidades do IAM](#) no Guia do usuário do IAM.

- Políticas de controle de serviço (SCPs) — SCPs são políticas JSON que especificam as permissões máximas para uma organização ou unidade organizacional (OU) em AWS Organizations. AWS Organizations é um serviço para agrupar e gerenciar centralmente vários Contas da AWS que sua empresa possui. Se você habilitar todos os recursos em uma organização, poderá aplicar políticas de controle de serviço (SCPs) a qualquer uma ou a todas as suas contas. O SCP limita as permissões para entidades nas contas dos membros, incluindo cada uma Usuário raiz da conta da AWS. Para obter mais informações sobre Organizations e SCPs, consulte [Políticas de controle de serviços](#) no Guia AWS Organizations do Usuário.
- Políticas de controle de recursos (RCPs) — RCPs são políticas JSON que você pode usar para definir o máximo de permissões disponíveis para recursos em suas contas sem atualizar as políticas do IAM anexadas a cada recurso que você possui. O RCP limita as permissões para recursos nas contas dos membros e pode afetar as permissões efetivas para identidades, incluindo a Usuário raiz da conta da AWS, independentemente de pertencerem à sua organização. Para obter mais informações sobre Organizations e RCPs, incluindo uma lista Serviços da AWS desse suporte RCPs, consulte [Políticas de controle de recursos \(RCPs\)](#) no Guia AWS Organizations do usuário.
- Políticas de sessão: são políticas avançadas que você transmite como um parâmetro quando cria de forma programática uma sessão temporária para um perfil ou um usuário federado. As permissões da sessão resultante são a interseção das políticas baseadas em identidade do usuário ou do perfil e das políticas de sessão. As permissões também podem ser provenientes de uma política baseada em recursos. Uma negação explícita em qualquer uma dessas políticas substitui a permissão. Para obter mais informações, consulte [Políticas de sessão](#) no Guia do usuário do IAM.

Vários tipos de política

Quando vários tipos de política são aplicáveis a uma solicitação, é mais complicado compreender as permissões resultantes. Para saber como AWS determinar se uma solicitação deve ser permitida quando vários tipos de políticas estão envolvidos, consulte [Lógica de avaliação de políticas](#) no Guia do usuário do IAM.

Como Amazon GameLift Servers funciona com o IAM

Antes de usar o IAM para gerenciar o acesso ao Amazon GameLift Servers, saiba quais recursos do IAM estão disponíveis para uso com Amazon GameLift Servers.

Recursos do IAM que você pode usar com Amazon GameLift Servers

Atributo do IAM	Amazon GameLift Servers Suporte
Políticas baseadas em identidade	Sim
Políticas baseadas em recurso	Não
Ações de políticas	Sim
Recursos de políticas	Sim
Chaves de condição de política (específicas do serviço)	Sim
ACLs	Não
ABAC (tags em políticas)	Sim
Credenciais temporárias	Sim
Permissões de entidade principal	Sim
Perfis de serviço	Sim
Perfis vinculados a serviço	Não

Para obter uma visão de alto nível de como Amazon GameLift Servers e outros AWS serviços funcionam com a maioria dos recursos do IAM, consulte [AWS serviços que funcionam com o IAM](#) no Guia do usuário do IAM.

Políticas baseadas em identidade para Amazon GameLift Servers

Compatível com políticas baseadas em identidade: sim

As políticas baseadas em identidade são documentos de políticas de permissões JSON que você pode anexar a uma identidade, como usuário do IAM, grupo de usuários ou perfil. Essas políticas controlam quais ações os usuários e perfis podem realizar, em quais recursos e em que condições. Para saber como criar uma política baseada em identidade, consulte [Definir permissões personalizadas do IAM com as políticas gerenciadas pelo cliente](#) no Guia do Usuário do IAM.

Com as políticas baseadas em identidade do IAM, é possível especificar ações e recursos permitidos ou negados, assim como as condições sob as quais as ações são permitidas ou negadas. Você não pode especificar a entidade principal em uma política baseada em identidade porque ela se aplica ao usuário ou perfil ao qual ela está anexada. Para saber mais sobre todos os elementos que podem ser usados em uma política JSON, consulte [Referência de elemento de política JSON do IAM](#) no Guia do usuário do IAM.

Exemplos de políticas baseadas em identidade para Amazon GameLift Servers

Para ver exemplos de Amazon GameLift Servers políticas baseadas em identidade, consulte.

[Exemplos de políticas baseadas em identidade para Amazon GameLift Servers](#)

Políticas baseadas em recursos dentro Amazon GameLift Servers

Compatibilidade com políticas baseadas em recursos: não

Políticas baseadas em recursos são documentos de políticas JSON que você anexa a um recurso. São exemplos de políticas baseadas em recursos as políticas de confiança de perfil do IAM e as políticas de bucket do Amazon S3. Em serviços compatíveis com políticas baseadas em recursos, os administradores de serviço podem usá-las para controlar o acesso a um recurso específico. Para o atributo ao qual a política está anexada, a política define quais ações uma entidade principal especificado pode executar nesse atributo e em que condições. Você deve [especificar uma entidade principal](#) em uma política baseada em recursos. Os diretores podem incluir contas, usuários, funções, usuários federados ou. Serviços da AWS

Para permitir o acesso entre contas, você pode especificar uma conta inteira ou as entidades do IAM em outra conta como a entidade principal em uma política baseada em recursos. Adicionar uma entidade principal entre contas à política baseada em recurso é apenas metade da tarefa de estabelecimento da relação de confiança. Quando o principal e o recurso são diferentes Contas da AWS, um administrador do IAM na conta confiável também deve conceder permissão à entidade principal (usuário ou função) para acessar o recurso. Eles concedem permissão ao anexar uma política baseada em identidade para a entidade. No entanto, se uma política baseada em recurso conceder acesso a uma entidade principal na mesma conta, nenhuma política baseada em identidade adicional será necessária. Consulte mais informações em [Acesso a recursos entre contas no IAM](#) no Guia do usuário do IAM.

Ações políticas para Amazon GameLift Servers

Compatível com ações de políticas: sim

Os administradores podem usar políticas AWS JSON para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos e em que condições.

O elemento `Action` de uma política JSON descreve as ações que podem ser usadas para permitir ou negar acesso em uma política. As ações de política geralmente têm o mesmo nome da operação de AWS API associada. Existem algumas exceções, como ações somente de permissão, que não têm uma operação de API correspondente. Algumas operações também exigem várias ações em uma política. Essas ações adicionais são chamadas de ações dependentes.

Incluem ações em uma política para conceder permissões para executar a operação associada.

Para uma lista de Amazon GameLift Servers ações, consulte [Ações definidas por Amazon GameLift Servers](#) na Referência de Autorização de Serviço.

Ações políticas em Amazon GameLift Servers use o seguinte prefixo antes da ação:

```
gamelift
```

Para especificar várias ações em uma única instrução, separe-as com vírgulas.

```
"Action": [  
    "gamelift:action1",  
    "gamelift:action2"  
]
```

Você também pode especificar várias ações usando caracteres-curinga (*). Por exemplo, para especificar todas as ações que começam com a palavra `Describe`, inclua a seguinte ação:

```
"Action": "gamelift:Describe*"
```

Para ver exemplos de Amazon GameLift Servers políticas baseadas em identidade, consulte [Exemplos de políticas baseadas em identidade para Amazon GameLift Servers](#)

Recursos políticos para Amazon GameLift Servers

Compatível com recursos de políticas: sim

Os administradores podem usar políticas AWS JSON para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos e em que condições.

O elemento de política JSON `Resource` especifica o objeto ou os objetos aos quais a ação se aplica. As instruções devem incluir um elemento `Resource` ou `NotResource`. Como prática recomendada, especifique um recurso usando seu [nome do recurso da Amazon \(ARN\)](#). Isso pode ser feito para ações que oferecem compatibilidade com um tipo de recurso específico, conhecido como permissões em nível de recurso.

Para ações que não oferecem compatibilidade com permissões em nível de recurso, como operações de listagem, use um curinga (*) para indicar que a instrução se aplica a todos os recursos.

```
"Resource": "*" 
```

Para uma lista de Amazon GameLift Servers tipos de recursos e seus ARNs, consulte [Recursos definidos por Amazon GameLift Servers](#) na Referência de Autorização de Serviço. Para saber com quais ações você pode especificar o ARN de cada recurso, consulte [Ações definidas por Amazon GameLift Servers](#).

Alguns Amazon GameLift Servers os recursos têm valores de ARN, o que permite que os recursos tenham seu acesso gerenciado usando políticas do IAM. A ferramenta Amazon GameLift Servers o recurso de frota tem um ARN com a seguinte sintaxe:

```
arn:${Partition}:gamelift:${Region}:${Account}:fleet/${FleetId}
```

Para obter mais informações sobre o formato de ARNs, consulte [Amazon Resource Names \(ARNs\)](#) no Referência geral da AWS.

Por exemplo, para especificar a frota `fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa` na instrução, use o seguinte ARN:

```
"Resource": "arn:aws:gamelift:us-west-2:123456789012:fleet/fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa" 
```

Para especificar todas as frotas que pertencem a uma conta específica, use o caractere curinga (*):

```
"Resource": "arn:aws:gamelift:us-west-2:123456789012:fleet/*" 
```

Para ver exemplos de Amazon GameLift Servers políticas baseadas em identidade, consulte.

[Exemplos de políticas baseadas em identidade para Amazon GameLift Servers](#)

Chaves de condição de política para Amazon GameLift Servers

Compatível com chaves de condição de política específicas de serviço: sim

Os administradores podem usar políticas AWS JSON para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos e em que condições.

O elemento Condition (ou bloco Condition) permite que você especifique condições nas quais uma instrução estiver em vigor. O elemento Condition é opcional. É possível criar expressões condicionais que usem [agentes de condição](#), como “igual a” ou “menor que”, para fazer a condição da política corresponder aos valores na solicitação.

Se você especificar vários elementos de Condition em uma declaração ou várias chaves em um único elemento de Condition, a AWS os avaliará usando uma operação lógica AND. Se você especificar vários valores para uma única chave de condição, AWS avalia a condição usando uma OR operação lógica. Todas as condições devem ser atendidas antes que as permissões da instrução sejam concedidas.

Você também pode usar variáveis de espaço reservado ao especificar condições. Por exemplo, é possível conceder a um usuário do IAM permissão para acessar um recurso somente se ele estiver marcado com seu nome de usuário do IAM. Para obter mais informações, consulte [Elementos da política do IAM: variáveis e tags](#) no Guia do usuário do IAM.

AWS suporta chaves de condição globais e chaves de condição específicas do serviço. Para ver todas as chaves de condição AWS globais, consulte as [chaves de contexto de condição AWS global](#) no Guia do usuário do IAM.

Para uma lista de Amazon GameLift Servers chaves de condição, consulte Chaves de [condição para Amazon GameLift Servers](#) na Referência de Autorização de Serviço. Para saber com quais ações e recursos você pode usar uma chave de condição, consulte [Ações definidas por Amazon GameLift Servers](#).

Para ver exemplos de Amazon GameLift Servers políticas baseadas em identidade, consulte.

[Exemplos de políticas baseadas em identidade para Amazon GameLift Servers](#)

ACLs em Amazon GameLift Servers

Suportes ACLs: Não

As listas de controle de acesso (ACLs) controlam quais diretores (membros da conta, usuários ou funções) têm permissões para acessar um recurso. ACLs são semelhantes às políticas baseadas em recursos, embora não usem o formato de documento de política JSON.

ABAC com Amazon GameLift Servers

Compatível com ABAC (tags em políticas): sim

O controle de acesso por atributo (ABAC) é uma estratégia de autorização que define as permissões com base em atributos. Em AWS, esses atributos são chamados de tags. Você pode anexar tags a entidades do IAM (usuários ou funções) e a vários AWS recursos. Marcar de entidades e atributos é a primeira etapa do ABAC. Em seguida, você cria políticas de ABAC para permitir operações quando a tag da entidade principal corresponder à tag do recurso que ela estiver tentando acessar.

O ABAC é útil em ambientes que estão crescendo rapidamente e ajuda em situações em que o gerenciamento de políticas se torna um problema.

Para controlar o acesso baseado em tags, forneça informações sobre as tags no [elemento de condição](#) de uma política usando as `aws:ResourceTag/key-name`, `aws:RequestTag/key-name` ou chaves de condição `aws:TagKeys`.

Se um serviço for compatível com as três chaves de condição para cada tipo de recurso, o valor será Sim para o serviço. Se um serviço for compatível com as três chaves de condição somente para alguns tipos de recursos, o valor será Parcial

Para obter mais informações sobre o ABAC, consulte [Definir permissões com autorização do ABAC](#) no Guia do usuário do IAM. Para visualizar um tutorial com etapas para configurar o ABAC, consulte [Usar controle de acesso baseado em atributos \(ABAC\)](#) no Guia do usuário do IAM.

Para obter um exemplo de política baseada em identidade que limita o acesso a um recurso com base em tags nesse recurso, consulte [Visualizar Amazon GameLift Servers frotas baseadas em tags](#).

Usando credenciais temporárias com Amazon GameLift Servers

Compatível com credenciais temporárias: sim

Alguns Serviços da AWS não funcionam quando você faz login usando credenciais temporárias. Para obter informações adicionais, incluindo quais Serviços da AWS funcionam com credenciais temporárias, consulte Serviços da AWS “[Trabalhe com o IAM](#)” no Guia do usuário do IAM.

Você está usando credenciais temporárias se fizer login AWS Management Console usando qualquer método, exceto um nome de usuário e senha. Por exemplo, quando você acessa AWS usando o link de login único (SSO) da sua empresa, esse processo cria automaticamente credenciais temporárias. Você também cria automaticamente credenciais temporárias quando faz login no console como usuário e, em seguida, alterna perfis. Para obter mais informações sobre como alternar funções, consulte [Alternar para um perfil do IAM \(console\)](#) no Guia do usuário do IAM.

Você pode criar manualmente credenciais temporárias usando a AWS API AWS CLI ou. Em seguida, você pode usar essas credenciais temporárias para acessar AWS. AWS recomenda que você gere credenciais temporárias dinamicamente em vez de usar chaves de acesso de longo prazo. Para obter mais informações, consulte [Credenciais de segurança temporárias no IAM](#).

Permissões principais entre serviços para Amazon GameLift Servers

Compatibilidade com o recurso de encaminhamento de sessões de acesso (FAS): sim

Quando você usa um usuário ou uma função do IAM para realizar ações AWS, você é considerado um principal. Ao usar alguns serviços, você pode executar uma ação que inicia outra ação em um serviço diferente. O FAS usa as permissões do diretor chamando um AWS service (Serviço da AWS), combinadas com a solicitação AWS service (Serviço da AWS) para fazer solicitações aos serviços posteriores. As solicitações do FAS são feitas somente quando um serviço recebe uma solicitação que requer interações com outros Serviços da AWS ou com recursos para ser concluída. Nesse caso, você precisa ter permissões para executar ambas as ações. Para obter detalhes da política ao fazer solicitações de FAS, consulte [Sessões de acesso direto](#).

Funções de serviço para Amazon GameLift Servers

Compatível com perfis de serviço: sim

O perfil de serviço é um [perfil do IAM](#) que um serviço assume para executar ações em seu nome. Um administrador do IAM pode criar, modificar e excluir um perfil de serviço do IAM. Para obter mais informações, consulte [Criar um perfil para delegar permissões a um AWS service \(Serviço da AWS\)](#) no Guia do Usuário do IAM.

Warning

A alteração das permissões de uma função de serviço pode falhar Amazon GameLift Servers funcionalidade. Edite funções de serviço somente quando Amazon GameLift Servers fornece orientação para fazer isso.

Permita que seu Amazon GameLift Servers-servidores de jogos hospedados para acessar outros AWS recursos, como uma AWS Lambda função ou um banco de dados do Amazon DynamoDB. Porque os servidores de jogos estão hospedados em frotas que Amazon GameLift Servers gerencia, você precisa de uma função de serviço que ofereça Amazon GameLift Servers acesso limitado aos seus outros AWS recursos. Para obter mais informações, consulte [Comunique-se com outros AWS recursos de suas frotas](#).

Funções vinculadas a serviços para Amazon GameLift Servers

Compatível com perfis vinculados ao serviço: Não

Uma função vinculada ao serviço é um tipo de função de serviço vinculada a um. AWS service (Serviço da AWS) O serviço pode presumir o perfil de executar uma ação em seu nome. As funções vinculadas ao serviço aparecem em você Conta da AWS e são de propriedade do serviço. Um administrador do IAM pode visualizar, mas não pode editar as permissões para perfis vinculados ao serviço.

Para obter detalhes sobre como criar ou gerenciar perfis vinculadas a serviços, consulte [AWS Serviços que funcionam com o IAM](#) no Guia do usuário do IAM.. Encontre um serviço na tabela que inclua um Yes na coluna Funções vinculadas ao serviço. Selecione Sim para visualizar a documentação do perfil vinculado ao serviço para esse serviço.

Exemplos de políticas baseadas em identidade para Amazon GameLift Servers

Por padrão, usuários e funções não têm permissão para criar ou modificar Amazon GameLift Servers recursos. Eles também não podem realizar tarefas usando a AWS API AWS Management Console, AWS Command Line Interface (AWS CLI) ou. Para conceder permissão aos usuários para executar ações nos recursos que eles precisam, um administrador do IAM pode criar políticas do IAM. O administrador pode então adicionar as políticas do IAM aos perfis e os usuários podem assumir os perfis.

Para aprender a criar uma política baseada em identidade do IAM ao usar esses documentos de política em JSON de exemplo, consulte [Criar políticas do IAM \(console\)](#) no Guia do usuário do IAM.

Para obter detalhes sobre ações e tipos de recursos definidos pelo Amazon GameLift Servers, incluindo o formato do ARNs para cada um dos tipos de recursos, consulte [Ações, recursos e chaves de condição para Amazon GameLift Servers](#) na Referência de Autorização de Serviço.

Tópicos

- [Práticas recomendadas de política](#)
- [Usar o Amazon GameLift Servers console](#)
- [Permitir que os usuários visualizem suas próprias permissões](#)
- [Permitir acesso de jogador para sessões de jogos](#)
- [Permitir acesso a um Amazon GameLift Servers queue \(fila\)](#)
- [Visualizar Amazon GameLift Servers frotas baseadas em tags](#)
- [Acesse um arquivo de compilação do jogo no Amazon S3](#)

Práticas recomendadas de política

As políticas baseadas em identidade determinam se alguém pode criar, acessar ou excluir Amazon GameLift Servers recursos em sua conta. Essas ações podem incorrer em custos para sua Conta da AWS. Ao criar ou editar políticas baseadas em identidade, siga estas diretrizes e recomendações:

- Comece com as políticas AWS gerenciadas e avance para as permissões de privilégios mínimos — Para começar a conceder permissões aos seus usuários e cargas de trabalho, use as políticas AWS gerenciadas que concedem permissões para muitos casos de uso comuns. Eles estão disponíveis no seu Conta da AWS. Recomendamos que você reduza ainda mais as permissões definindo políticas gerenciadas pelo AWS cliente que sejam específicas para seus casos de uso. Para obter mais informações, consulte [Políticas gerenciadas pela AWS](#) ou [Políticas gerenciadas pela AWS para funções de trabalho](#) no Guia do usuário do IAM.
- Aplique permissões de privilégio mínimo: ao definir permissões com as políticas do IAM, conceda apenas as permissões necessárias para executar uma tarefa. Você faz isso definindo as ações que podem ser executadas em recursos específicos sob condições específicas, também conhecidas como permissões de privilégio mínimo. Para obter mais informações sobre como usar o IAM para aplicar permissões, consulte [Políticas e permissões no IAM](#) no Guia do usuário do IAM.
- Use condições nas políticas do IAM para restringir ainda mais o acesso: você pode adicionar uma condição às políticas para limitar o acesso a ações e recursos. Por exemplo, você pode escrever uma condição de política para especificar que todas as solicitações devem ser enviadas usando SSL. Você também pode usar condições para conceder acesso às ações de serviço se elas forem usadas por meio de uma ação específica AWS service (Serviço da AWS), como AWS CloudFormation. Para obter mais informações, consulte [Elementos da política JSON do IAM: condição](#) no Guia do usuário do IAM.
- Use o IAM Access Analyzer para validar suas políticas do IAM a fim de garantir permissões seguras e funcionais: o IAM Access Analyzer valida as políticas novas e existentes para que elas

sigam a linguagem de política do IAM (JSON) e as práticas recomendadas do IAM. O IAM Access Analyzer oferece mais de cem verificações de política e recomendações práticas para ajudar a criar políticas seguras e funcionais. Para obter mais informações, consulte [Validação de políticas do IAM Access Analyzer](#) no Guia do Usuário do IAM.

- Exigir autenticação multifator (MFA) — Se você tiver um cenário que exija usuários do IAM ou um usuário root, ative Conta da AWS a MFA para obter segurança adicional. Para exigir MFA quando as operações de API forem chamadas, adicione condições de MFA às suas políticas. Para obter mais informações, consulte [Configuração de acesso à API protegido por MFA](#) no Guia do Usuário do IAM.

Para obter mais informações sobre as práticas recomendadas do IAM, consulte [Práticas recomendadas de segurança no IAM](#) no Guia do usuário do IAM.

Usar o Amazon GameLift Servers console

Para acessar o Amazon GameLift Servers console, você deve ter um conjunto mínimo de permissões. Essas permissões devem permitir que você liste e visualize detalhes sobre o Amazon GameLift Servers recursos em seu Conta da AWS. Caso crie uma política baseada em identidade mais restritiva que as permissões mínimas necessárias, o console não funcionará como pretendido para entidades (usuários ou perfis) com essa política.

Para garantir que essas entidades ainda possam usar o Amazon GameLift Servers console, adicione permissões a usuários e grupos com a sintaxe nos exemplos a seguir e em [Exemplos de permissões administrativas](#). Para obter mais informações, consulte [Definir permissões de usuário para Amazon GameLift Servers](#).

Usuários que trabalham com Amazon GameLift Servers as operações AWS CLI por meio de nossa AWS API não exigem permissões mínimas do console. Em vez disso, é possível limitar o acesso somente às operações que o usuário precisa executar. Por exemplo, um usuário jogador, agindo em nome de clientes do jogo, exige acesso para solicitar sessões de jogo, colocar jogadores em jogos e outras tarefas.

Para obter informações sobre as permissões necessárias para usar todos Amazon GameLift Servers recursos do console, consulte a sintaxe de permissões para administradores em [Exemplos de permissões administrativas](#)

Permitir que os usuários visualizem suas próprias permissões

Este exemplo mostra como criar uma política que permita que os usuários do IAM visualizem as políticas gerenciadas e em linha anexadas a sua identidade de usuário. Essa política inclui permissões para concluir essa ação no console ou programaticamente usando a API AWS CLI ou 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": "*"
    }
  ]
}
```

Permitir acesso de jogador para sessões de jogos

Para colocar jogadores em sessões de jogo, os clientes de jogos e os serviços de back-end precisam de permissões. Para obter exemplos de políticas para esses cenários, consulte [Exemplos de permissão de usuário do jogador](#).

Permitir acesso a um Amazon GameLift Servers queue (fila)

O exemplo a seguir fornece ao usuário acesso a uma Amazon GameLift Servers filas.

Essa política concede ao usuário permissões para adicionar, atualizar e excluir destinos de fila com as seguintes ações: `gamelift:UpdateGameSessionQueue`, `gamelift>DeleteGameSessionQueue` e `gamelift:DescribeGameSessionQueues`. Conforme mostrado, essa política usa o elemento `Resource` para limitar o acesso a uma única fila: `gamesessionqueue/examplequeue123`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ViewSpecificQueueInfo",
      "Effect": "Allow",
      "Action": [
        "gamelift:DescribeGameSessionQueues"
      ],
      "Resource": "arn:aws:gamelift::gamesessionqueue/examplequeue123"
    },
    {
      "Sid": "ManageSpecificQueue",
      "Effect": "Allow",
      "Action": [
        "gamelift:UpdateGameSessionQueue",
        "gamelift>DeleteGameSessionQueue"
      ],
      "Resource": "arn:aws:gamelift::gamesessionqueue/examplequeue123"
    }
  ]
}
```

Visualizar Amazon GameLift Servers frotas baseadas em tags

Você pode usar condições em sua política baseada em identidade para controlar o acesso a Amazon GameLift Servers recursos baseados em tags. Este exemplo mostra como você pode criar uma política que permita visualizar uma frota se a tag `Owner` corresponder ao nome do usuário. Essa política também concede as permissões necessárias concluir essa operação no console.

```
{
  "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": {"gamelift:ResourceTag/Owner": "${aws:username}"}
      }
    }
  ]
}
```

Acesse um arquivo de compilação do jogo no Amazon S3

Depois de integrar seu servidor de jogos com Amazon GameLift Servers, faça o upload dos arquivos de compilação para o Amazon S3. Para Amazon GameLift Servers para acessar os arquivos de compilação, use a política a seguir.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion"
      ]
    }
  ]
}
```



```
    ],
    "Resource": "arn:aws:s3:::bucket-name/object-name"
  }
]
```

Para obter mais informações sobre o upload Amazon GameLift Servers arquivos do jogo, veja [Implemente uma criação de servidor personalizada para Amazon GameLift Servers hospedagem](#).

Solução de problemas Amazon GameLift Servers identidade e acesso

Use as informações a seguir para ajudá-lo a diagnosticar e corrigir problemas comuns que você pode encontrar ao trabalhar com Amazon GameLift Servers e AWS Identity and Access Management (IAM).

Tópicos

- [Não estou autorizado a realizar uma ação em Amazon GameLift Servers](#)
- [Não estou autorizado a realizar iam: PassRole](#)
- [Quero permitir que pessoas fora da minha Conta da AWS acessem meu Amazon GameLift Servers recursos](#)

Não estou autorizado a realizar uma ação em Amazon GameLift Servers

Se isso AWS Management Console indicar que você não está autorizado a realizar uma ação, entre em contato com o administrador AWS da conta para obter ajuda. Caso seu administrador seja a pessoa que forneceu suas credenciais de início de sessão.

O erro de exemplo a seguir ocorre quando o usuário mateojackson do IAM tenta usar o console para visualizar detalhes sobre uma fila, mas não tem as permissões `gamelift:DescribeGameSessionQueues`.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
gamelift:DescribeGameSessionQueues on resource: examplequeue123
```

Neste caso, Mateo pede ao administrador para atualizar suas políticas para permitir a ele o acesso de leitura ao recurso `examplequeue123` usando a ação `gamelift:DescribeGameSessionQueues`.

Não estou autorizado a realizar `iam:PassRole`

Se você receber um erro informando que não está autorizado a realizar a `iam:PassRole` ação, suas políticas devem ser atualizadas para permitir que você passe uma função para Amazon GameLift Servers.

Alguns Serviços da AWS permitem que você passe uma função existente para esse serviço em vez de criar uma nova função de serviço ou uma função vinculada ao serviço. Para fazer isso, é preciso ter permissões para passar o perfil para o serviço.

O exemplo de erro a seguir ocorre quando um usuário do IAM chamado `marymajor` tenta usar o console para realizar uma ação no Amazon GameLift Servers. No entanto, a ação exige que o serviço tenha permissões concedidas por uma função de serviço. Mary não tem permissões para passar o perfil para o serviço.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
iam:PassRole
```

Nesse caso, as políticas de Mary devem ser atualizadas para permitir que ela realize a ação `iam:PassRole`.

Se precisar de ajuda, entre em contato com seu AWS administrador. Seu administrador é a pessoa que forneceu suas credenciais de login.

Quero permitir que pessoas fora da minha Conta da AWS acessem meu Amazon GameLift Servers recursos

É possível criar um perfil que os usuários de outras contas ou pessoas fora da sua organização podem usar para acessar seus recursos. É possível especificar quem é confiável para assumir o perfil. Para serviços que oferecem suporte a políticas baseadas em recursos ou listas de controle de acesso (ACLs), você pode usar essas políticas para conceder às pessoas acesso aos seus recursos.

Para saber mais, consulte:

- Para saber se Amazon GameLift Servers suporta esses recursos, consulte [Como Amazon GameLift Servers funciona com o IAM](#).

- Para saber como fornecer acesso aos seus recursos em todos os Contas da AWS que você possui, consulte Como [fornecer acesso a um usuário do IAM em outro Conta da AWS que você possui](#) no Guia do usuário do IAM.
- Para saber como fornecer acesso aos seus recursos a terceiros Contas da AWS, consulte Como [fornecer acesso Contas da AWS a terceiros](#) no Guia do usuário do IAM.
- Para saber como conceder acesso por meio da federação de identidades, consulte [Conceder acesso a usuários autenticados externamente \(federação de identidades\)](#) no Guia do usuário do IAM.
- Para conhecer a diferença entre perfis e políticas baseadas em recurso para acesso entre contas, consulte [Acesso a recursos entre contas no IAM](#) no Guia do usuário do IAM.

AWS políticas gerenciadas para Amazon GameLift Servers

Uma política AWS gerenciada é uma política autônoma criada e administrada por AWS. AWS as políticas gerenciadas são projetadas para fornecer permissões para muitos casos de uso comuns, para que você possa começar a atribuir permissões a usuários, grupos e funções.

Lembre-se de que as políticas AWS gerenciadas podem não conceder permissões de privilégio mínimo para seus casos de uso específicos porque elas estão disponíveis para uso de todos os AWS clientes. Recomendamos que você reduza ainda mais as permissões definindo as [políticas gerenciadas pelo cliente](#) que são específicas para seus casos de uso.

Você não pode alterar as permissões definidas nas políticas AWS gerenciadas. Se AWS atualizar as permissões definidas em uma política AWS gerenciada, a atualização afetará todas as identidades principais (usuários, grupos e funções) às quais a política está anexada. AWS é mais provável que atualize uma política AWS gerenciada quando uma nova AWS service (Serviço da AWS) for lançada ou novas operações de API forem disponibilizadas para serviços existentes.

Para obter mais informações, consulte [Políticas gerenciadas pela AWS](#) no Guia do usuário do IAM.

AWS política gerenciada: GameLiftContainerFleetPolicy

Você pode se associar GameLiftContainerFleetPolicy às suas funções do IAM.

A política concede permissões para ações de computação em um Amazon GameLift Servers frota de contêineres. Uma frota de contêineres é um conjunto de recursos de hospedagem que Amazon GameLift Servers gerencia para você. Amazon GameLift Servers precisa de permissões para se conectar ao Amazon GameLift Servers serviço e outros AWS serviços em seu nome.

Ao criar uma frota de contêineres com Amazon GameLift Servers, forneça uma função de serviço do IAM com a política `GameLiftContainerFleetPolicy` gerenciada anexada. Para obter instruções sobre como criar a função de serviço, consulte [Configure uma função de serviço do IAM para Amazon GameLift Servers](#).

Para obter mais informações, consulte [GameLiftContainerFleetPolicy](#).

Detalhes das permissões

Esta política inclui as seguintes permissões.

- `cloudwatch`— Permite Amazon GameLift Servers para gravar registros de sessões de jogos em um stream de registros de CloudWatch eventos da Amazon em sua AWS conta.
- `cloudwatch` — Permite Amazon GameLift Servers para criar um grupo de CloudWatch registros para organizar os dados da sessão do jogo no fluxo de registros.
- `s3`— Permite Amazon GameLift Servers para gravar registros de sessões de jogos em um bucket do Amazon Simple Storage Service em sua AWS conta.
- `s3`— Permite Amazon GameLift Servers para recuperar o Região da AWS local em que um bucket específico do Amazon S3 reside, usando a ação da API. `s3:GetBucketLocation`
- `gamelift` — Permite Amazon GameLift Servers para recuperar um token de autenticação que permite que um servidor de jogo hospedado se comunique com o Amazon GameLift Servers serviço por meio de sua AWS conta.

Amazon GameLift Servers atualizações nas políticas AWS gerenciadas

Exibir detalhes sobre atualizações nas políticas AWS gerenciadas para Amazon GameLift Servers desde que esse serviço começou a rastrear essas mudanças. Para receber alertas automáticos sobre alterações nesta página, assine o feed RSS no [Amazon GameLift Servers Página de notas de lançamento](#).

Alteração	Descrição	Data
GameLiftContainerFleetPolicy : alteração	Amazon GameLift Servers adicionou novas permissões para recuperar o Região da AWS de um bucket do Amazon S3.	5 de fevereiro de 2024

Alteração	Descrição	Data
GameLiftContainerFleetPolicy – Nova política	Amazon GameLift Servers adicionou novas permissões para permitir que os contêineres do servidor de jogos sejam executados Amazon GameLift Servers frotas gerenciadas.	12 de novembro de 2024
Amazon GameLift Servers começou a rastrear as alterações	Amazon GameLift Servers começou a rastrear as mudanças em suas políticas AWS gerenciadas.	12 de novembro de 2024

Registro e monitoramento com Amazon GameLift Servers

O monitoramento é uma parte importante da manutenção da confiabilidade, disponibilidade e desempenho do Amazon GameLift Servers e suas AWS soluções. Você deve coletar dados de monitoramento de todas as partes da sua AWS solução para poder depurar com mais facilidade uma falha de vários pontos, caso ocorra.

AWS e Amazon GameLift Servers forneça várias ferramentas para monitorar seus recursos de hospedagem de jogos e responder a possíveis incidentes.

CloudWatch Alarmes da Amazon

Usando CloudWatch os alarmes da Amazon, você assiste a uma única métrica durante um período de tempo especificado por você. Se a métrica exceder um determinado limite, uma notificação será enviada para um tópico do Amazon SNS ou para uma política de Auto AWS Scaling. CloudWatch os alarmes são acionados quando seu estado muda e são mantidos por um determinado número de períodos, não por estarem em um estado específico. Para obter mais informações, consulte [Monitore Amazon GameLift Servers com a Amazon CloudWatch](#).

AWS CloudTrail Registros

CloudTrail fornece um registro das ações realizadas por um usuário, função ou AWS serviço no Amazon GameLift Servers. Usando as informações coletadas por CloudTrail, você pode determinar a solicitação que foi feita para Amazon GameLift Servers, o endereço IP a partir do qual

a solicitação foi feita, quem fez a solicitação, quando ela foi feita e detalhes adicionais. Para obter mais informações, consulte [Registro em log Amazon GameLift Servers Chamadas de API com AWS CloudTrail](#).

Validação de conformidade para Amazon GameLift Servers

Amazon GameLift Servers não está no escopo de nenhum programa de AWS conformidade.

Para saber se um AWS service (Serviço da AWS) está dentro do escopo de programas de conformidade específicos, consulte [Serviços da AWS Escopo por Programa de Conformidade](#) [Serviços da AWS](#) e escolha o programa de conformidade em que você está interessado. Para obter informações gerais, consulte Programas de [AWS conformidade Programas AWS](#) de .

Você pode baixar relatórios de auditoria de terceiros usando AWS Artifact. Para obter mais informações, consulte [Baixar relatórios em AWS Artifact](#) .

Sua responsabilidade de conformidade ao usar Serviços da AWS é determinada pela confidencialidade de seus dados, pelos objetivos de conformidade de sua empresa e pelas leis e regulamentações aplicáveis. AWS fornece os seguintes recursos para ajudar na conformidade:

- [Governança e conformidade de segurança](#): esses guias de implementação de solução abordam considerações sobre a arquitetura e fornecem etapas para implantar recursos de segurança e conformidade.
- [Referência de serviços qualificados para HIPAA](#): lista os serviços qualificados para HIPAA. Nem todos Serviços da AWS são elegíveis para a HIPAA.
- AWS Recursos de <https://aws.amazon.com/compliance/resources/> de conformidade — Essa coleção de pastas de trabalho e guias pode ser aplicada ao seu setor e local.
- [AWS Guias de conformidade do cliente](#) — Entenda o modelo de responsabilidade compartilhada sob a ótica da conformidade. Os guias resumem as melhores práticas de proteção Serviços da AWS e mapeiam as diretrizes para controles de segurança em várias estruturas (incluindo o Instituto Nacional de Padrões e Tecnologia (NIST), o Conselho de Padrões de Segurança do Setor de Cartões de Pagamento (PCI) e a Organização Internacional de Padronização (ISO)).
- [Avaliação de recursos com regras](#) no Guia do AWS Config desenvolvedor — O AWS Config serviço avalia o quão bem suas configurações de recursos estão em conformidade com as práticas internas, as diretrizes e os regulamentos do setor.
- [AWS Security Hub](#) — Isso AWS service (Serviço da AWS) fornece uma visão abrangente do seu estado de segurança interno AWS. O Security Hub usa controles de segurança para avaliar os

recursos da AWS e verificar a conformidade com os padrões e as práticas recomendadas do setor de segurança. Para obter uma lista dos serviços e controles aceitos, consulte a [Referência de controles do Security Hub](#).

- [Amazon GuardDuty](#) — Isso AWS service (Serviço da AWS) detecta possíveis ameaças às suas cargas de trabalho Contas da AWS, contêineres e dados monitorando seu ambiente em busca de atividades suspeitas e maliciosas. GuardDuty pode ajudá-lo a atender a vários requisitos de conformidade, como o PCI DSS, atendendo aos requisitos de detecção de intrusões exigidos por determinadas estruturas de conformidade.
- [AWS Audit Manager](#) — Isso AWS service (Serviço da AWS) ajuda você a auditar continuamente seu AWS uso para simplificar a forma como você gerencia o risco e a conformidade com as regulamentações e os padrões do setor.

Resiliência em Amazon GameLift Servers

Se você estiver usando Amazon GameLift Servers FleetIQ como um recurso independente da Amazon EC2, consulte [Segurança na Amazon EC2 no](#) Guia do EC2 usuário da Amazon.

A infraestrutura AWS global é construída em torno de AWS regiões e zonas de disponibilidade. AWS As regiões fornecem várias zonas de disponibilidade fisicamente separadas e isoladas, conectadas a redes de baixa latência, alta taxa de transferência e alta redundância. Com as zonas de disponibilidade, é possível projetar e operar aplicações e bancos de dados que automaticamente executam o failover entre as zonas sem interrupção. As zonas de disponibilidade são altamente disponíveis, tolerantes a falhas e escaláveis que uma ou várias infraestruturas de data center tradicionais.

Para obter mais informações sobre AWS regiões e zonas de disponibilidade, consulte [infraestrutura AWS global](#).

Além da infraestrutura AWS global, Amazon GameLift Servers oferece os seguintes recursos para ajudar a atender às suas necessidades de resiliência de dados:

- Filas multirregionais — Amazon GameLift Servers as filas de sessões de jogo são usadas para colocar novas sessões de jogo com os recursos de hospedagem disponíveis. As filas que abrangem várias regiões podem redirecionar colocações em sessões de jogos no caso de uma interrupção regional. Para obter mais informações e as melhores práticas sobre a criação de filas de sessões de jogos, consulte [Personalize uma fila de sessões de jogo](#).

- Escalonamento automático da capacidade — mantenha a integridade e a disponibilidade de seus recursos de hospedagem usando Amazon GameLift Servers ferramentas de dimensionamento. Essas ferramentas oferecem uma variedade de opções que permitem ajustar a capacidade da frota de acordo com as necessidades do jogo e dos jogadores. Para ter mais informações sobre a escalabilidade, consulte [Escalando a capacidade de hospedagem de jogos com Amazon GameLift Servers](#).
- Distribuição entre instâncias — Amazon GameLift Servers distribui o tráfego de entrada em várias instâncias, dependendo do tamanho da frota. Como melhor prática, os jogos em produção devem ter várias instâncias para manter a disponibilidade caso uma instância não esteja íntegra ou não responda.
- Armazenamento Amazon S3 — compilações e scripts de servidores de jogos que são enviados para Amazon GameLift Servers são armazenados no Amazon S3 usando a classe de armazenamento Standard, que usa várias replicações de data center para aumentar a resiliência. Os logs de sessão de jogo também são armazenados no Amazon S3 usando a classe de armazenamento Standard.

Segurança da infraestrutura em Amazon GameLift Servers

Se você estiver usando Amazon GameLift Servers FleetIQ como um recurso independente da Amazon EC2, consulte [Segurança na Amazon EC2 no](#) Guia do EC2 usuário da Amazon.

Como um serviço gerenciado, Amazon GameLift Servers é protegido pelos procedimentos AWS globais de segurança de rede descritos no whitepaper [Amazon Web Services: Visão geral dos processos de segurança](#).

Você usa chamadas de API AWS publicadas para acessar Amazon GameLift Servers através da rede. Os clientes devem oferecer suporte a Transport Layer Security (TLS) 1.2 ou posterior. Recomendamos usar o TLS 1.3 ou posterior. Os clientes também devem ter compatibilidade com conjuntos de criptografia com perfect forward secrecy (PFS) como Ephemeral Diffie-Hellman (DHE) ou Ephemeral Elliptic Curve Diffie-Hellman (ECDHE). A maioria dos sistemas modernos como Java 7 e versões posteriores oferece compatibilidade com esses modos.

Além disso, as solicitações devem ser assinadas usando um ID da chave de acesso e uma chave de acesso secreta associada a uma entidade principal do IAM. Ou é possível usar o [AWS Security Token Service](#) (AWS STS) para gerar credenciais de segurança temporárias para assinar solicitações.

A ferramenta Amazon GameLift Servers O serviço coloca todas as frotas nas nuvens privadas virtuais da Amazon (VPCs) para que cada frota exista em uma área logicamente isolada na AWS nuvem. Você pode usar: Amazon GameLift Servers políticas para controlar o acesso de endpoints de VPC específicos ou específicos. VPCs Efetivamente, isso isola o acesso à rede a um determinado Amazon GameLift Servers recurso somente da VPC específica dentro da AWS rede. Ao criar uma frota, você especifica um intervalo de números de porta e endereços IP. Esses intervalos limitam como o tráfego de entrada pode acessar servidores de jogos hospedados em uma VPC da frota. Use as melhores práticas de segurança padrão ao selecionar as configurações de acesso à frota.

Análise de configuração e vulnerabilidade em Amazon GameLift Servers

Se você estiver usando Amazon GameLift Servers FleetIQ como um recurso independente da Amazon EC2, consulte [Segurança na Amazon EC2 no](#) Guia do EC2 usuário da Amazon.

A configuração e os controles de TI são uma responsabilidade compartilhada entre a AWS e você, nosso cliente. Para obter mais informações, consulte o [modelo de responsabilidade AWS compartilhada](#). AWS lida com tarefas básicas de segurança, como sistema operacional (SO) convidado e aplicação de patches em bancos de dados, configuração de firewall e recuperação de desastres. Esses procedimentos foram revisados e certificados por terceiros certificados. Para obter mais detalhes, consulte o seguinte recurso: [Amazon Web Services: visão geral dos processos de segurança](#) (whitepaper).

As práticas recomendadas de segurança a seguir também abordam a configuração e a análise de vulnerabilidades no Amazon GameLift Servers:

- Os clientes são responsáveis pelo gerenciamento do software implantado em Amazon GameLift Servers instâncias para hospedagem de jogos. Especificamente:
 - O software do aplicativo de servidor de jogos fornecido pelo cliente deve ser mantido, incluindo atualizações e patches de segurança. Para atualizar o software do servidor de jogos, faça o upload de uma nova versão para Amazon GameLift Servers, crie uma nova frota para ele e redirecione o tráfego para a nova frota.
 - A imagem de máquina da Amazon (AMI) base, que inclui o sistema operacional, é atualizada somente quando uma frota é criada. Para aplicar patches, atualizar e proteger o sistema operacional e outros aplicativos que fazem parte da AMI, recicle as frotas regularmente, independentemente das atualizações do servidor de jogos.

- Os clientes devem considerar a atualização regular de seus jogos com as versões mais recentes do SDK, incluindo o AWS SDK, o Amazon GameLift Servers SDK do servidor e o Amazon GameLift Servers SDK de cliente para servidores em tempo real.

Melhores práticas de segurança do Amazon GameLift Servers

Se você estiver usando Amazon GameLift Servers FleetIQ como um recurso independente com a Amazon EC2, consulte [Segurança na Amazon EC2 no](#) Guia do EC2 usuário da Amazon.

O Amazon GameLift Servers oferece uma série de recursos de segurança a serem considerados no desenvolvimento e na implementação das suas próprias políticas de segurança. As melhores práticas a seguir são diretrizes gerais e não representam uma solução completa de segurança. Como essas práticas recomendadas podem não ser adequadas ou suficientes para o seu ambiente, trate-as como considerações úteis em vez de prescrições.

Não abra portas para a internet

É altamente recomendável não abrir portas para a internet, pois isso representa um risco à segurança. Por exemplo, se você usa [UpdateFleetPortSettings](#) para abrir uma porta de desktop remoto como esta:

```
{
  "FleetId": "<fleet identifier>",
  "InboundPermissionAuthorizations": [
    {
      "FromPort": 3389,
      "IpRange": "0.0.0.0/0",
      "Protocol": "RDP",
      "ToPort": 3389
    }
  ]
}
```

Você está permitindo que qualquer pessoa na internet acesse a instância.

Em vez disso, abra a porta com um endereço IP específico ou um intervalo de endereços. Por exemplo, semelhante a este:

```
{
```

```
"FleetId": "<fleet identifier>",
"InboundPermissionAuthorizations": [
  {
    "FromPort": 3389,
    "IpRange": "54.186.139.221/32",
    "Protocol": "TCP",
    "ToPort": 3389
  }
]
```

Saiba mais

Para obter mais informações sobre como você pode tornar seu uso Amazon GameLift Servers mais seguro, consulte o [pilar AWS Well-Architected Tool Segurança](#).

Amazon GameLift Servers guias de referência

Esta seção contém documentação de referência sobre como usar o Amazon GameLift Servers.

Tópicos

- [API de serviço para Amazon GameLift Servers](#)
- [SDK de servidor 5.x para Amazon GameLift Servers](#)
- [SDK do servidor para Amazon GameLift Servers versão 4 e anteriores](#)
- [Amazon GameLift Servers Amazon GameLift Servers referência Realtime](#)
- [Eventos de posicionamento de sessões de jogo](#)
- [Amazon GameLift Servers Versões da AMI](#)
- [Endpoints e cotas do Amazon GameLift Servers](#)
- [Amazon GameLift Servers Sinalizadores de ping UDP](#)

API de serviço para Amazon GameLift Servers

Use essa lista baseada em tarefas para encontrar operações de API ao criar suas soluções de hospedagem de Amazon GameLift Servers jogos e outros recursos. O AWS SDK inclui essas operações no `aws.gamelift` namespace. [Baixe o AWS SDK](#) ou [veja a documentação de referência Amazon GameLift Servers da API](#). Você também pode usar a API com a interface de linha de AWS comando (AWS CLI), conforme documentado na [referência do AWS CLI comando](#).

A API inclui dois conjuntos de operações para hospedagem gerenciada de jogos:

- [Gerenciar recursos Amazon GameLift Servers de hospedagem](#)
- [Inicie sessões de jogo e junte-se aos jogadores](#)

A API Amazon GameLift Servers de serviço também contém operações para uso com outras Amazon GameLift Servers ferramentas e soluções. Para ver uma lista de FleetIQ APIs, consulte [Operações de FleetIQ API](#). Para obter uma lista de FlexMatch APIs para matchmaking, veja [operações de FlexMatch API](#).

Gerenciar recursos Amazon GameLift Servers de hospedagem

Chame essas operações para configurar recursos de hospedagem para os servidores de jogos, escalar a capacidade para atender à demanda dos jogadores, obter métricas de desempenho e utilização e muito mais. Use essas operações de API ao hospedar servidores de jogos com Amazon GameLift Servers, inclusive Amazon GameLift ServersRealtime. Você também pode trabalhar no [Amazon GameLift Serversconsole](#) para a maioria das tarefas de gerenciamento de recursos ou pode fazer chamadas com a ferramenta AWS Command Line Interface (AWS CLI).

Preparar servidores de jogos para implantação

Carregue e configure o código do servidor de jogos do jogo em preparação para implantação e lançamento em recursos de hospedagem.

Gerenciar compilações do servidor de jogos personalizado

- [upload-build](#) — Carregue arquivos de compilação de um caminho local e crie um novo Amazon GameLift Servers recurso de compilação. Essa operação, disponível como um AWS CLI comando, é a forma mais comum de fazer upload de compilações de servidores de jogos.
- [CreateBuild](#) — Crie uma nova compilação usando arquivos armazenados em um bucket do Amazon S3.
- [ListBuilds](#) — Obtenha uma lista de todas as construções enviadas para uma Amazon GameLift Servers região.
- [DescribeBuild](#) — Recupere informações associadas a uma construção.
- [UpdateBuild](#) — Altere os metadados da compilação, incluindo nome e versão da compilação.
- [DeleteBuild](#) — Remover uma compilação do Amazon GameLift Servers.

Gerenciar scripts Amazon GameLift Servers Realtime de configuração

- [CreateScript](#) — Faça upload de JavaScript arquivos e crie um novo recurso de Amazon GameLift Servers script.
- [ListScripts](#) — Obtenha uma lista de todos os Realtime scripts enviados para uma Amazon GameLift Servers região.
- [DescribeScript](#) — Recupere informações associadas a um Realtime script.
- [UpdateScript](#) — Altere os metadados do script e faça o upload do conteúdo revisado do script.

- [DeleteScript](#)— Remover um Realtime script do Amazon GameLift Servers.

Configurar recursos de computação para hospedagem

Configure os recursos de hospedagem e implante-os com o script de criação ou Realtime configuração do seu servidor de jogos.

Criar e gerenciar frotas

- [CreateFleet](#)— Configure e implante uma nova Amazon GameLift Servers frota de recursos de computação para executar seus servidores de jogos. Depois de implantados, os servidores de jogos são automaticamente lançados conforme configurados e prontos para hospedar sessões de jogo.
- [ListFleets](#)— Obtenha uma lista de todas as frotas em uma Amazon GameLift Servers região.
- [DeleteFleet](#)— Remova uma frota que não está mais executando servidores de jogos nem hospedando jogadores.
- Visualize/atualize as localizações da frota.
 - [CreateFleetLocations](#)— Adicione locais remotos a uma frota existente que suporta vários locais
 - [DescribeFleetLocationAttributes](#)— Obtenha uma lista de todos os locais remotos de uma frota e veja o status atual de cada local.
 - [DeleteFleetLocations](#)— Remova locais remotos de uma frota que suporta vários locais.
- Visualize/atualize as configurações de frota.
 - [DescribeFleetAttributes/UpdateFleetAttributes](#)— Visualize ou altere os metadados e as configurações de uma frota para proteção da sessão de jogo e limites de criação de recursos.
 - [DescribeFleetPortSettings/UpdateFleetPortSettings](#)— Visualize ou altere as permissões de entrada (endereço IP e intervalos de configuração de porta) permitidas para uma frota.
 - [DescribeRuntimeConfiguration/UpdateRuntimeConfiguration](#)— Visualize ou altere quais processos do servidor (e quantos) devem ser executados em cada instância de uma frota.

Gerenciar capacidade da frota

- [Describe EC2 InstanceLimits](#) — Recupere o número máximo de instâncias permitidas para a AWS conta atual e o nível de uso atual.
- [DescribeFleetCapacity](#)— Recupere as configurações de capacidade atuais da região de origem de uma frota.

- [DescribeFleetLocationCapacity](#)— Recupere as configurações de capacidade atuais para cada local de uma frota com vários locais.
- [UpdateFleetCapacity](#)— Ajuste manualmente as configurações de capacidade de uma frota.
- Configure o :
 - [PutScalingPolicy](#)— ative o escalonamento automático baseado em metas, crie uma política de escalonamento automático personalizada ou atualize uma política existente.
 - [DescribeScalingPolicies](#)— Recupere uma política de auto-scaling existente.
 - [DeleteScalingPolicy](#)— exclua uma política de auto-scaling e impeça que ela afete a capacidade da frota.
 - [StartFleetActions](#)— Reinicie as políticas de auto-scaling de uma frota.
 - [StopFleetActions](#)— Suspender as políticas de auto-escalonamento de uma frota.

Monitore a atividade da frota.

- [DescribeFleetUtilization](#)— Recupere estatísticas sobre o número de processos do servidor, sessões de jogo e jogadores que estão atualmente ativos em uma frota.
- [DescribeFleetLocationUtilization](#)— Recupere estatísticas de utilização de cada local em uma frota com vários locais.
- [DescribeFleetEvents](#)— Visualize os eventos registrados de uma frota durante um período de tempo especificado.
- [DescribeGameSessions](#)— Recupere os metadados da sessão do jogo, incluindo o tempo de execução do jogo e a contagem atual de jogadores.

Configurar filas para o posicionamento de sessões de jogo

Configure as filas de várias frotas e várias regiões para posicionar sessões de jogo com os melhores recursos de hospedagem disponíveis para custo, latência e resiliência.

- [CreateGameSessionQueue](#)— Crie uma fila para uso ao processar solicitações de colocação em sessões de jogos.
- [DescribeGameSessionQueues](#)— Recupere filas de sessões de jogo definidas em uma Amazon GameLift Servers região.
- [UpdateGameSessionQueue](#)— Alterar a configuração de uma fila de sessões de jogo.
- [DeleteGameSessionQueue](#)— Remova uma fila de sessões de jogo da região.

Gerenciar aliases

Use aliases para representar suas frotas ou crie um destino alternativo de terminal. Os aliases são úteis ao fazer a transição da atividade do jogo de uma frota para outra, como durante as atualizações de compilação do servidor de jogos.

- [CreateAlias](#)— Defina um novo alias e, opcionalmente, atribua-o a uma frota.
- [ListAliases](#)— Obtenha todos os aliases de frota definidos em uma Amazon GameLift Servers região.
- [DescribeAlias](#)— Recupere informações sobre um alias existente.
- [UpdateAlias](#)— Altere as configurações de um alias, como redirecioná-lo de uma frota para outra.
- [DeleteAlias](#)— Remova um alias da região.
- [ResolveAlias](#)— Obtenha o ID da frota para o qual um alias especificado aponta.

Conectar-se às instâncias de hospedagem gerenciada

Visualize informações sobre instâncias individuais em uma frota ou solicite acesso remoto a uma instância de frota especificada para solução de problemas.

- [DescribeInstances](#)— Obtenha informações sobre cada instância em uma frota, incluindo ID da instância, endereço IP, localização e status.
- [GetInstanceAccess](#)— Solicite as credenciais de acesso necessárias para se conectar remotamente a uma instância específica em uma frota.

Configurar o emparelhamento da VPC

Crie e gerencie conexões de emparelhamento de VPC entre seus recursos de Amazon GameLift Servers hospedagem e outros recursos. AWS

- [CreateVpcPeeringAuthorization](#)— Autorize uma conexão de emparelhamento com um dos seus VPCs
- [DescribeVpcPeeringAuthorizations](#)— Recupere autorizações de conexão de peering válidas.
- [DeleteVpcPeeringAuthorization](#)— Exclua uma autorização de conexão de emparelhamento.
- [CreateVpcPeeringConnection](#)— Estabeleça uma conexão de emparelhamento entre o VPC de Amazon GameLift Servers uma frota e uma de suas VPCs

- [DescribeVpcPeeringConnections](#)— recupere informações sobre conexões de emparelhamento de VPC ativas ou pendentes com uma frota. Amazon GameLift Servers
- [DeleteVpcPeeringConnection](#)— Exclua uma conexão de emparelhamento de VPC com uma frota. Amazon GameLift Servers

Inicie sessões de jogo e junte-se aos jogadores

Chame essas operações por meio de um serviço de backend para iniciar novas sessões de jogo, obter informações sobre sessões de jogo existentes e adicionar jogadores às sessões de jogo. Essas operações são para uso com servidores de jogos personalizados hospedados em Amazon GameLift Servers. Se você estiver usando Amazon GameLift ServersRealtime, gerencie as sessões de jogo usando [Amazon GameLift ServersRealtime referência da API do cliente \(C#\)](#) o.

- Inicie novas sessões de jogo para um ou mais jogadores.
 - [StartGameSessionPlacement](#)— Peça Amazon GameLift Servers para encontrar os melhores recursos de hospedagem disponíveis e iniciar uma nova sessão de jogo. Esse é o método preferido para criar novas sessões de jogo. Ele se baseia em filas de sessões de jogo para rastrear a disponibilidade de hospedagem em várias regiões e usa FleetIQ algoritmos para priorizar posicionamentos com base na latência do jogador, custo de hospedagem, localização etc.
 - [DescribeGameSessionPlacement](#)— Obtenha detalhes e status de uma solicitação de colocação.
 - [StopGameSessionPlacement](#)— Cancelar uma solicitação de colocação.
 - [CreateGameSession](#)— Inicie uma nova sessão de jogo vazia em um local específico da frota. Essa operação oferece maior controle sobre por onde começar a sessão do jogo, em vez de usá-la FleetIQ para avaliar as opções de posicionamento. Você deve adicionar jogadores à nova sessão de jogo em uma etapa separada.
- Coloque jogadores em sessões de jogo existentes. Encontre sessões de jogo em execução com slots de jogadores disponíveis e reserve-os para novos jogadores.
 - [CreatePlayerSession](#)— Reserve um espaço aberto para um jogador participar de uma sessão de jogo.
 - [CreatePlayerSessions](#)— Reserve vagas abertas para vários jogadores participarem de uma sessão de jogo.
- Trabalhe com os dados da sessão de jogo e da sessão de jogadores. Gerencie informações sobre sessões de jogo e sessões de jogadores.

- [SearchGameSessions](#)— Solicite uma lista de sessões de jogo ativas com base em um conjunto de critérios de pesquisa.
- [DescribeGameSessions](#)— Recupere metadados para sessões de jogo específicas, incluindo o tempo de atividade e a contagem atual de jogadores.
- [DescribeGameSessionDetails](#)— Recupere metadados, incluindo a configuração de proteção da sessão de jogo, para uma ou mais sessões de jogo.
- [DescribePlayerSessions](#)— Obtenha detalhes sobre a atividade do jogador, incluindo status, tempo de jogo e dados do jogador.
- [UpdateGameSession](#)— Altere as configurações da sessão do jogo, como contagem máxima de jogadores e política de ingresso.
- [GetGameSessionLogUrl](#)— Obtenha a localização dos registros salvos para uma sessão de jogo.

SDK de servidor 5.x para Amazon GameLift Servers

Esta seção fornece documentação de referência para o SDK 5.x do servidor para Amazon GameLift Servers. O SDK do servidor fornece as principais funcionalidades que seu servidor de jogos usa para interagir com o Amazon GameLift Servers serviço. Por exemplo, o servidor de jogos recebe solicitações do serviço para iniciar e interromper sessões de jogo e fornece atualizações regulares sobre o status das sessões de jogo ao serviço. Integre os servidores de jogos ao SDK do servidor antes de implantá-los para hospedagem.

Use essa referência do SDK do servidor para integrar seus servidores de jogos multijogador personalizados para hospedagem com Amazon GameLift Servers. Para obter orientação sobre o processo de integração, consulte [Adicione Amazon GameLift Servers ao seu servidor de jogos](#).

A versão principal mais recente do SDK do servidor Amazon GameLift Servers é a 5.x. Os seguintes recursos de hospedagem exigem o uso da versão 5.x:

- Amazon GameLift Servers Em qualquer lugar
- Amazon GameLift Servers plugin para Unreal Engine e Unity

Tópicos

- [Atualizações no SDK 5 do servidor para Amazon GameLift Servers](#)
- [Migre para o SDK 5.x do servidor para Amazon GameLift Servers](#)
- [SDK 5.x do servidor C++ para Amazon GameLift Servers -- Ações](#)

- [SDK 5.x do servidor C# para Amazon GameLift Servers -- Ações](#)
- [SDK do servidor Go para Amazon GameLift Servers -- Ações](#)
- [SDK 5.x do servidor C++ \(Unreal\) para -- Ações Amazon GameLift Servers](#)

Atualizações no SDK 5 do servidor para Amazon GameLift Servers

Seus servidores de jogos hospedados usam o SDK do servidor Amazon GameLift Servers para se comunicar com o Amazon GameLift Servers serviço para iniciar e gerenciar sessões de jogo para jogadores. A versão mais recente, SDK 5 do Amazon GameLift Servers servidor, oferece várias melhorias e suporte para novos Amazon GameLift Servers recursos. Se a versão do seu servidor de jogos atualmente usa o SDK de Amazon GameLift Servers servidor 4 ou anterior, siga as orientações neste tópico para atualizar seus jogos.

Amazon GameLift ServersA versão 5.0.0 e superior do SDK do servidor inclui as seguintes atualizações:

- Linguagens expandidas: as bibliotecas estão disponíveis nas seguintes linguagens: C++, C# e Go. Você pode compilar as bibliotecas C++ para uso com o Unreal Engine.
- Suporte a plug-ins de mecanismos de jogos — Os plug-ins Amazon GameLift Servers autônomos para Unreal Engine e Unity exigem bibliotecas SDK 5 de Amazon GameLift Servers servidor. Esses plug-ins oferecem fluxos de trabalho guiados para integrar, testar e implantar seus jogos para Amazon GameLift Servers hospedagem. Consulte a documentação [Amazon GameLift Serversplugin para Unity \(servidor SDK 5.x\)](#) e [Amazon GameLift Serversplugin para Unreal Engine](#).
- Amazon GameLift ServersSuporte em qualquer lugar — Com o Anywhere Fleets, você pode configurar seus próprios recursos de hospedagem para usar Amazon GameLift Servers recursos (incluindo matchmaking). Adicione o Amazon GameLift Servers Agente para automatizar o gerenciamento do ciclo de vida das sessões de jogo. Use frotas Anywhere para hospedagem de produção com hardware local ou configure ambientes de teste para desenvolvimento rápido e iterativo de jogos. [Hospedagem Anywhere](#)See e o [Amazon GameLift ServersAgente](#).
- Ferramentas de teste atualizadas — O recurso Amazon GameLift Servers Anywhere permite que você configure ambientes de teste locais ou baseados na nuvem para seus jogos. Configure o teste com ou sem o Amazon GameLift Servers agente. Essas ferramentas substituem o Amazon GameLift Servers Local. Consulte [Configure testes locais com o Amazon GameLift Servers Anywhere](#).

- Solução .NET consolidada para C#: o SDK 5.1+ do servidor C# oferece suporte ao .NET Framework 4.6.2 (atualizado da versão 4.6.1) e ao .NET 6.0 em uma única solução. O .NET Standard 2.1 está disponível com as bibliotecas criadas pelo Unity.
- Novo recurso Compute: esse novo recurso combina diferentes tipos de recurso de hospedagem. Inclui recursos de hospedagem baseados em nuvem (frotas gerenciadas EC2 ou de contêineres) e recursos de hospedagem controlados pelo cliente (frotas em qualquer lugar). Inclui as seguintes atualizações:
 - As novas chamadas de API para o Compute recurso incluem: [ListCompute\(\)](#), [DescribeCompute\(\)](#) e [GetComputeAccess\(\)](#). Essas ações retornam informações de recursos de hospedagem para qualquer tipo de Amazon GameLift Servers frota. Em geral, para frotas com servidores de jogos que usam o SDK 5.x do servidor, use as ações específicas da computação para substituir as ações específicas da instância. Além disso, essas ações devem ser usadas em frotas de Anywhere sem o Amazon GameLift Servers Agente: [RegisterCompute\(\)](#), [DeregisterCompute\(\)](#) e [GetComputeAuthToken\(\)](#).
 - Nova métrica ActiveCompute com CloudWatch dimensões FleetIdLocation, ComputeType e. Essa métrica substitui a métrica anterior ActiveInstances.
- Amazon EC2 Systems Manager (SSM) para acesso remoto — Para maior segurança, use SSM em vez de SSH ao se conectar a instâncias em Amazon GameLift Servers frotas gerenciadas. Consulte [Conecte-se remotamente a Amazon GameLift Servers instâncias de frota](#).

Migre para o SDK 5.x do servidor para Amazon GameLift Servers

Para atualizar um projeto de jogo para usar o SDK versão 5.x do servidor, faça as seguintes alterações:

1. Obtenha o SDK de servidor mais recente para Amazon GameLift Servers pacote para seu ambiente de desenvolvimento [[site de download](#)]. Siga as instruções de instalação no arquivo Readme para o pacote e a versão baixados. Veja essas instruções para usar o servidor SDKs com seu projeto de jogo.
 - [Para ambientes de desenvolvimento que usam C++, C# ou Go](#)
 - [Para projetos do Unreal Engine \(somente bibliotecas do SDK do servidor C++ para Unreal\)](#)
 - [Para projetos do Unity \(somente bibliotecas do SDK do servidor C# para Unity\)](#)
 - [Para uso com o Amazon GameLift Servers plugin do Unreal Engine](#)
 - [Para uso com o Amazon GameLift Servers plugin para Unity](#)

2. Atualize o código do servidor da seguinte forma:

- Altere a função de retorno de chamada do código do servidor de `onCreateGameSession()` para `onStartGameSession()`.
- Atualize as entradas `InitSDK()` conforme apropriado:
 - Se você planeja implantar a versão do servidor de jogo em uma EC2 frota Amazon GameLift Servers gerenciada ou em uma frota Anywhere com o Amazon GameLift Servers Agente:

Chame `InitSDK()` sem parâmetros ([C++](#)) ([C#](#)) ([Unreal](#)). Essa chamada configura o ambiente computacional e uma WebSocket conexão com o Amazon GameLift Servers serviço.

- Se você planeja implantar o servidor de jogo construído em uma frota Anywhere sem o Amazon GameLift Servers Agente:

Chame `InitSDK()` com parâmetros do servidor ([C++](#)) ([C#](#)) ([Unreal](#)). Um processo de servidor de jogos usa esses parâmetros para estabelecer uma conexão com o Amazon GameLift Servers serviço.

3. Se a compilação do servidor de jogos ou outras aplicações hospedadas se comunicarem com outros recursos da AWS durante a execução, será necessário alterar a forma como a aplicação obtém acesso a esses recursos. Substitua o uso de `AssumeRoleCredentials` pela nova ação do SDK do servidor `GetFleetRoleCredentials()` (para servidores de jogos) ou use credenciais compartilhadas (para outros aplicativos). Para obter mais informações sobre como implementar essa alteração, consulte [Comunique-se com outros AWS recursos de suas frotas](#).
4. Se o projeto chamou a ação `GetInstanceCertificate()` do SDK do servidor para recuperar um certificado TLS, modifique o código para usar o novo `GetComputeCertificate()` ([C++](#)) ([C#](#)) ([Unreal](#)).
5. Ao fazer o upload da compilação do jogo para Amazon GameLift Servers (como com [upload-build](#) ou [CreateBuild\(\)](#)), defina o `ServerSdkVersion` parâmetro para a versão 5.x que você está usando (esse parâmetro atualmente é 4.0.2). Esse parâmetro deve corresponder às bibliotecas reais do SDK do servidor na compilação do servidor de jogos. Se você especificar a versão errada para uma compilação de servidor de jogos carregada, todas as frotas criadas com essa compilação falharão. Consulte [Implemente uma criação de servidor personalizada para Amazon GameLift Servers hospedagem](#).

O exemplo a seguir ilustra como especificar a versão do SDK do servidor:

```
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 você usa scripts para se conectar remotamente a frotas gerenciadas, atualize os scripts para usar o novo processo, conforme descrito em [Conecte-se remotamente a Amazon GameLift Servers instâncias de frota](#).

SDK 5.x do servidor C++ para Amazon GameLift Servers -- Ações

Use a referência 5.x do SDK do servidor para integrar seu jogo multijogador para hospedagem com. Amazon GameLift Servers Para obter orientação sobre o processo de integração, consulte [Adicione Amazon GameLift Servers ao seu servidor de jogos](#).

Note

Este tópico descreve a API Amazon GameLift Servers C++ que você pode usar ao criar com a Biblioteca Padrão C++ (`std`). Especificamente, essa documentação se aplica ao código que você compila com a opção `-DDGAMELIFT_USE_STD=1`.

SDK 5.x do servidor C++ para Amazon GameLift Servers -- Tipos de dados

Use a referência 5.x do SDK do servidor Amazon GameLift Servers C++ para integrar seu jogo multijogador para hospedagem com. Amazon GameLift Servers Para obter orientação sobre o processo de integração, consulte [Adicione Amazon GameLift Servers ao seu servidor de jogos](#).

Note

Este tópico descreve a API Amazon GameLift Servers C++ que você pode usar ao criar com a Biblioteca Padrão C++ (`std`). Especificamente, essa documentação se aplica ao código que você compila com a opção `-DDGAMELIFT_USE_STD=1`.

SDK 5.x do servidor C++ para Amazon GameLift Servers -- Ações

Tipos de dados

- [LogParameters](#)
- [ProcessParameters](#)
- [UpdateGameSession](#)
- [GameSession](#)
- [ServerParameters](#)
- [StartMatchBackfillRequest](#)
- [Jogador](#)
- [DescribePlayerSessionsRequest](#)
- [StopMatchBackfillRequest](#)
- [AttributeValue](#)
- [GetFleetRoleCredentialsRequest](#)
- [AwsLongOutcome](#)
- [AwsStringOutcome](#)
- [DescribePlayerSessionsOutcome](#)
- [DescribePlayerSessionsResult](#)
- [GenericOutcome](#)
- [GenericOutcomeCallable](#)
- [PlayerSession](#)
- [StartMatchBackfillOutcome](#)
- [StartMatchBackfillResult](#)
- [GetComputeCertificateOutcome](#)
- [GetComputeCertificateResult](#)
- [GetFleetRoleCredentialsOutcome](#)
- [GetFleetRoleCredentialsResult](#)
- [Inicializar SDKOutcome](#)
- [GameLiftError](#)
- [Enumerações](#)

LogParameters

Um objeto que identifica arquivos gerados durante uma sessão de jogo que você Amazon GameLift Servers deseja carregar e armazenar após o término da sessão de jogo. O servidor do jogo fornece LogParameters o Amazon GameLift Servers como parte de um ProcessParameters objeto em uma [ProcessReady\(\)](#) chamada.

Properties	Descrição
LogPaths	A lista de caminhos de diretório para os arquivos de log do servidor de jogos que você Amazon GameLift Servers deseja armazenar para acesso futuro. O processo do servidor gera esses arquivos durante cada sessão de jogo. Você define caminhos e nomes de arquivos em seu servidor de jogo e os armazena no diretório raiz de construção do jogo. Os caminhos de log devem ser absolutos. Por exemplo, se a compilação do jogo armazenar logs de sessão de jogo em um caminho como <code>MyGame\sessionLogs\</code>, o caminho estará <code>c:\game\MyGame\sessionLogs</code> em uma instância do Windows. Digite: <code>std::vector<std::string></code> Obrigatório: não

ProcessParameters

Esse tipo de dados contém o conjunto de parâmetros enviados Amazon GameLift Servers em [ProcessReady\(\)](#) a.

Properties	Descrição

LogParameters	Um objeto com caminhos de diretório para arquivos que são gerados durante uma sessão de jogo. Amazon GameLift Servers copia e armazena os arquivos para acesso futuro. Digite: <code>Aws::GameLift::Server:: LogParameters</code> Obrigatório: não
OnHealthCheck	A função de retorno de chamada que Amazon GameLift Servers invoca para solicitar um relatório de status de saúde do processo do servidor. Amazon GameLift Servers chama essa função a cada 60 segundos e espera 60 segundos por uma resposta. O processo do servidor retorna TRUE se estiver íntegro, FALSE se não estiver íntegro. Se nenhuma resposta for retornada, Amazon GameLift Servers registra o processo do servidor como não íntegro. Digite: <code>std::function<bool()> onHealthCheck</code> Obrigatório: não

OnProcessTerminate	A função de retorno de chamada que Amazon GameLift Servers invoca para forçar o encerramento do processo do servidor. Depois de chamar essa função, Amazon GameLift Servers aguarda 5 minutos até que o processo do servidor seja encerrado e responda com uma ProcessEnding() chamada antes de encerrar o processo do servidor. Digite: <code>std::function<void()> onProcessTerminate</code> Obrigatório: Sim
OnStartGameSession	A função de retorno de chamada que Amazon GameLift Servers invoca para ativar uma nova sessão de jogo. Amazon GameLift Servers chama essa função em resposta a uma solicitação do cliente CreateGameSession. A função de retorno de chamada transmite um objeto GameSession. Digite: <code>const std::function<void (Aws::GameLift::Model::GameSession)> onStartGameSession</code> Obrigatório: Sim

OnUpdateGameSession	A função de retorno de chamada que Amazon GameLift Servers invoca para passar um objeto de sessão de jogo atualizado para o processo do servidor. Amazon GameLift Servers chama essa função quando uma solicitação de preenchimento de correspondência é processada para fornecer dados atualizados do matchmaker. Ele transmite um objeto GameSession, uma atualização de status (updateReason) e o ID do tíquete de alocação de correspondência. Digite: <code>std::function<void(Aws::GameLift::Server::Model::UpdateGameSession)> onUpdateGameSession</code> Obrigatório: não
Porta	O número da porta em que o processo de servidor escuta para novas conexões de jogador. O valor deve estar dentro do intervalo de portas configurado para qualquer frota que implanta essa compilação do servidor de jogo. Esse número de porta está incluído nos objetos das sessões de jogo e jogador, que as sessões de jogo usam ao se conectar a um processo de servidor. Digite: Integer Obrigatório: Sim

UpdateGameSession

Esse tipo de dados é atualizado para um objeto de sessão de jogo, o que inclui o motivo pelo qual a sessão de jogo foi atualizada e o ID do tíquete de alocação relacionada, se a alocação for usada para preencher as sessões dos jogadores na sessão do jogo.

Propriedades	Descrição
GameSession	Um objeto GameSession. O objeto <code>GameSession</code> contém propriedades que descrevem uma sessão de jogo. Digite: <code>Aws::GameLift::Server::GameSession</code> Obrigatório: Sim
UpdateReason	O motivo pelo qual a sessão do jogo está sendo atualizada. Digite: <code>Aws::GameLift::Server::UpdateReason</code> Obrigatório: Sim
BackfillTicketId	O ID do tíquete de alocação que está tentando atualizar a sessão do jogo. Digite: <code>std::string</code> Obrigatório: não

GameSession

Esse tipo de dados fornece detalhes de uma sessão de jogo.

Propriedades	Descrição
GameSessionId	Um identificador exclusivo da sessão de jogo. Um ARN de sessão de jogo tem o

Propriedades	Descrição
	seguinte formato: <code>arn:aws:gamelift:<region>::gamesession/<fleet ID>/<custom ID string or idempotency token></code> . Digite: <code>std::string</code> Obrigatório: não
Name	Um rótulo descritivo da sessão do jogo. Digite: <code>std::string</code> Obrigatório: não
FleetId	Um identificador exclusivo da frota em que a sessão de jogo está sendo executada. Digite: <code>std::string</code> Obrigatório: não
MaximumPlayerSessionCount	O número máximo de conexões do jogador com a sessão do jogo. Digite: <code>int</code> Obrigatório: não
Porta	O número da porta da sessão de jogo. Para se conectar a um servidor de Amazon GameLift Servers jogos, um aplicativo precisa do endereço IP e do número da porta. Digite: <code>int</code> Obrigatório: não

Propriedades	Descrição
IpAddress	O endereço IP da sessão de jogo. Para se conectar a um servidor de Amazon GameLift Servers jogos, um aplicativo precisa do endereço IP e do número da porta. Digite: <code>std::string</code> Obrigatório: não
GameSessionData	Um conjunto de propriedades de sessão de jogo personalizadas, formatadas como um único valor de string. Digite: <code>std::string</code> Obrigatório: não
MatchmakerData	Informações sobre o processo de matchmaking usado para criar a sessão do jogo, na sintaxe JSON, formatado como uma string. Além da configuração de marcação de jogos usada, ele contém dados sobre todos os jogadores atribuídos à partida, incluindo atributos do jogador e atribuições da equipe. Digite: <code>std::string</code> Obrigatório: não

Propriedades	Descrição
GameProperties	Um conjunto de propriedades personalizadas para uma sessão de jogo, formatado como pares de chave:valor. Essas propriedades são passadas com uma solicitação de início de uma nova sessão de jogo. Digite: <code>std :: vector < GameProperty ></code> Obrigatório: não
DnsName	O identificador DNS atribuído à instância que está executando a sessão do jogo. Os valores têm o formato a seguir: • Frotas habilitadas para TLS: <code><unique identifier>.<region identifier>.amazongamelift.com</code> . • Non-TLS-enabled frotas: <code>ec2-<unique identifier>.compute.amazonaws.com</code> . Ao se conectar a uma sessão de jogo que está sendo executada em uma frota compatível com TLS, você deve usar o nome DNS, não o endereço IP. Digite: <code>std::string</code> Obrigatório: não

ServerParameters

Informações que um processo de servidor de jogos usa para estabelecer uma conexão com o Amazon GameLift Servers serviço. Inclua esses parâmetros ao chamar [InitSDK\(\)](#) somente se a versão do servidor do jogo for implantada em uma frota Anywhere ou em uma frota de contêineres

sem o Amazon GameLift Servers Agente. Para todos os outros cenários de implantação, chame [InitSDK\(\)](#) sem parâmetros.

Propriedades	Descrição
webSocketUrl	Os GameLiftServerSdkEndpoint Amazon GameLift Servers retornos quando você RegisterCompute compra um recurso de computação Amazon GameLift Servers Anywhere. Digite: std::string Obrigatório: Sim
processId	Um identificador exclusivo registrado no processo do servidor que hospeda seu jogo. Digite: std::string Obrigatório: Sim
hostId	O HostID é o ComputeName usado quando você registrou seu computador. Para obter mais informações, consulte, RegisterCompute. Digite: std::string Obrigatório: Sim
fleetId	O identificador exclusivo da frota na qual o computador está registrado. Para obter mais informações, consulte, RegisterCompute. Digite: std::string Obrigatório: Sim
authToken	O token de autenticação gerado por Amazon GameLift Servers isso autentica seu servidor

Propriedades	Descrição
	em Amazon GameLift Servers. Para obter mais informações, consulte, GetComputeAuthToken .
	Digite: <code>std::string</code>
	Obrigatório: Sim

StartMatchBackfillRequest

Informações usadas para criar uma solicitação de alocação de marcação de jogos. O servidor do jogo comunica essas informações Amazon GameLift Servers em uma [StartMatchBackfill\(\)](#) chamada.

Propriedades	Descrição
GameSessionArn	Um identificador da sessão de jogo exclusivo . A operação de API retorna GetGameSessionId o identificador no formato de ARN. Digite: <code>std::string</code> Obrigatório: Sim
MatchmakingConfigurationArn	um identificador exclusivo, no formato de ARN, para o marcador de jogos a ser usado para essa solicitação. O marcador de jogos ARN da sessão de jogo original está no objeto de sessão de jogo na propriedade de dados do marcador de jogos. Saiba mais sobre os dados do marcador de jogos em Trabalhar com dados do marcador de jogos . Digite: <code>std::string</code> Obrigatório: Sim
Jogadores	Um conjunto de dados que representa todos os jogadores que estão na sessão do jogo.

Propriedades	Descrição
	O marcador de jogos usa essas informações para pesquisar novos jogadores que sejam correspondências ideais para os jogadores atuais. Digite: <code>std::vector<Player></code> Obrigatório: Sim
TicketId	Um identificador exclusivo de um tíquete de solicitação de marcação de jogo ou de alocação de correspondência. Se você não fornecer um valor, Amazon GameLift Servers gera um. Use esse identificador para acompanhar o status do tíquete de alocação de correspondência ou cancelar a solicitação, se necessário. Digite: <code>std::string</code> Obrigatório: não

Jogador

Esse tipo de dados representa um jogador em marcação de jogos. Ao iniciar uma solicitação de matchmaking, um jogador tem um ID de jogador, atributos, e possivelmente dados de latência. Amazon GameLift Servers adiciona informações da equipe após a realização de uma partida.

Propriedades	Descrição
LatencyInMS	Um conjunto de valores expressos em milissegundos que indicam a quantidade de latência que um jogador experimenta quando conectado a um local. Se essa propriedade for usada, o player só será correspondido aos locais listados. Se um

Propriedades	Descrição
	marcador de jogos tiver uma regra que avalie a latência do jogador, os jogadores deverão reportar a latência a ser correspondida. Digite: <code>Dictionary<string,int></code> Obrigatório: não
PlayerAttributes	Uma coleção de pares de chave:valor contendo informações de jogadores para uso em marcação de jogos. As chaves de atributos do jogador devem corresponder às PlayerAttributes usadas em um conjunto de regras de matchmaking. Para obter mais informações sobre os atributos do jogador, consulte AttributeValue. Digite: <code>std::map<std::string, AttributeValue></code> Obrigatório: não
PlayerId	Um identificador exclusivo para um jogador. Digite: <code>std::string</code> Obrigatório: não
Equipe	O nome da equipe à qual o jogador está designado em uma partida. Você define o nome da equipe no conjunto de regras de marcação de jogos. Digite: <code>std::string</code> Obrigatório: não

DescribePlayerSessionsRequest

Um objeto que especifica quais sessões de jogador devem ser recuperadas. O processo do servidor fornece essas informações com uma [DescribePlayerSessions\(\)](#) chamada para Amazon GameLift Servers.

Propriedades	Descrição
GameSessionId	Um identificador da sessão de jogo exclusivo . Use esse parâmetro para solicitar todas as sessões de jogador da sessão de jogo especificada. O formato do ID da sessão do jogo é <code>arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string></code>. O GameSessionID é uma string de ID personalizada ou uma Digite: <code>std::string</code> Obrigatório: não
PlayerSessionId	Um identificador exclusivo para uma sessão de jogador. Use esse parâmetro para solicitar uma única sessão de jogador específica. Digite: <code>std::string</code> Obrigatório: não
PlayerId	Um identificador exclusivo para um jogador. Use este parâmetro para solicitar todas as sessões de jogador para um jogador específico. Consulte Gerar jogador IDs. Digite: <code>std::string</code> Obrigatório: não

Propriedades	Descrição
PlayerSessionStatusFilter	O status da sessão de jogador para filtrar resultados. Entre os status da sessão de jogador possíveis incluem: • RESERVED - A solicitação de sessão do jogador foi recebida, mas o jogador não se conectou ao processo do servidor nem foi validado. • ACTIVE - O jogador foi validado pelo processo do servidor e está conectado. • COMPLETED – A conexão do jogador foi interrompida. • TIMEDOUT – Uma solicitação de sessão de jogador foi recebida, mas o jogador não se conectou ou não foi validado dentro do tempo limite (60 segundos). Digite: <code>std::string</code> Obrigatório: não
NextToken	O token que indica o início da próxima página de resultados. Para especificar o início do conjunto de resultados, não forneça um valor. Se fornecer um ID de sessão do jogador, esse parâmetro será ignorado. Digite: <code>std::string</code> Obrigatório: não

Propriedades	Descrição
Limite	O número máximo de resultados a serem retornados. Se fornecer um ID de sessão do jogador, esse parâmetro será ignorado. Digite: <code>int</code> Obrigatório: não

StopMatchBackfillRequest

Informações usadas para cancelar uma solicitação de alocação de marcação de jogos. O servidor do jogo comunica essas informações ao Amazon GameLift Servers serviço em uma [StopMatchBackfill\(\)](#) chamada.

Propriedades	Descrição
GameSessionArn	Um identificador exclusivo da sessão de jogo da solicitação que está sendo cancelada. Digite: <code>char[]</code> Obrigatório: não
MatchmakingConfigurationArn	Um identificador exclusivo do marcador de jogos para o qual essa solicitação foi enviada. Digite: <code>char[]</code> Obrigatório: não
TicketId	Um identificador exclusivo do tíquete de solicitação de alocação a ser cancelado. Digite: <code>char[]</code> Obrigatório: não

AttributeValue

Use esses valores em pares de chave-valor de atributo [Jogador](#). Esse objeto permite especificar um valor de atributo usando qualquer um dos tipos de dados válidos: string, número, matriz de string ou mapa de dados. Cada objeto `AttributeValue` deve usar exatamente uma das propriedades disponíveis: S, N, SL ou SDM.

Propriedades	Descrição
AttrType	Especifica o tipo de valor do atributo. Os possíveis tipos de valor de atributo incluem: • NONE • STRING • DOUBLE • STRING_LIST • STRING_DOUBLE_MAP Obrigatório: não
S	Representa um valor de atributo de string. Digite: <code>std::string</code> Obrigatório: não
N	Representa um valor de atributo numérico. Digite: <code>double</code> Obrigatório: não
SL	Representa uma matriz de valores de atributos de string. Digite: <code>std::vector<std::string></code> Obrigatório: não
SDM	Representa um dicionário de chaves de string e valores duplos.

Propriedades	Descrição
	Digite: <code>std::map<std::string, double></code>
	Obrigatório: não

GetFleetRoleCredentialsRequest

Esse tipo de dados dá ao servidor do jogo acesso limitado aos seus outros AWS recursos. Para obter mais informações, consulte, [Configure uma função de serviço do IAM para Amazon GameLift Servers](#).

Propriedades	Descrição
RoleArn	O Amazon Resource Name (ARN) da função de serviço que estende o acesso limitado aos seus AWS recursos. Digite: <code>std::string</code> Obrigatório: não
RoleSessionName	O nome da sessão da função que você pode usar para identificar uma AWS Security Token Service AssumeRole sessão de forma exclusiva. Esse nome é exposto em registros de auditoria, como os de CloudTrail. Digite: <code>std::string</code> Obrigatório: não

AwsLongOutcome

Esse tipo de dados resulta de uma ação e produz um objeto com as seguintes propriedades:

Propriedades	Descrição
Resultado	O resultado da ação.

Propriedades	Descrição
	Digite: <code>long</code> Obrigatório: não
ResultWithOwnership	O resultado da ação, convertido como um rvalue, para que o código de chamada possa se apropriar do objeto. Digite: <code>long&&</code> Obrigatório: não
Bem-sucedida	Se a ação foi bem-sucedida ou não. Digite: <code>bool</code> Obrigatório: Sim
Erro	O erro que ocorreu se a ação não foi bem-sucedida. Digite: the section called “GameLiftError” Obrigatório: não

AwsStringOutcome

Esse tipo de dados resulta de uma ação e produz um objeto com as seguintes propriedades:

Propriedades	Descrição
Resultado	O resultado da ação. Digite: <code>std::string</code> Obrigatório: não
ResultWithOwnership	O resultado da ação, convertido como um rvalue, para que o código de chamada possa se apropriar do objeto. Digite: <code>long&&</code>

Propriedades	Descrição
	Obrigatório: não
Bem-sucedida	Se a ação foi bem-sucedida ou não. Digite: <code>bool</code> Obrigatório: Sim
Erro	O erro que ocorreu se a ação não foi bem-sucedida. Digite: the section called “GameLiftError” Obrigatório: não

DescribePlayerSessionsOutcome

Esse tipo de dados resulta de uma ação e produz um objeto com as seguintes propriedades:

Propriedades	Descrição
Resultado	O resultado da ação. Digite: the section called “DescribePlayerSessionsResult” Obrigatório: não
ResultWithOwnership	O resultado da ação, convertido como um rvalue, para que o código de chamada possa se apropriar do objeto. Digite: <code>Aws::GameLift::Server::Model::DescribePlayerSessionsResult&&</code> Obrigatório: não
Bem-sucedida	Se a ação foi bem-sucedida ou não. Digite: <code>bool</code> Obrigatório: Sim

Propriedades	Descrição
Erro	O erro que ocorreu se a ação não foi bem-sucedida. Digite: the section called “GameLiftError” Obrigatório: não

DescribePlayerSessionsResult

Uma coleção de objetos contendo propriedades para cada sessão de jogador que corresponde à solicitação.

Propriedades	Descrição
NextToken	Um token que indica o início da próxima página sequencial de resultados. Use o token retornado com uma chamada anterior para essa operação. Para começar no início do conjunto de resultados, não especifique um valor. Se um ID de sessão de jogador for especificado, esse parâmetro será ignorado. Digite: <code>std::string</code> Obrigatório: Sim
PlayerSessions	Digite: <code>IList<the section called “PlayerSession”></code> Obrigatório:
ResultWithOwnership	O resultado da ação, convertido como um rvalue, para que o código de chamada possa se apropriar do objeto. Digite: <code>std::string&&</code> Obrigatório: não
Bem-sucedida	Se a ação foi bem-sucedida ou não. Digite: <code>bool</code>

Propriedades	Descrição
	Obrigatório: Sim
Erro	O erro que ocorreu se a ação não foi bem-sucedida. Digite: the section called “GameLiftError” Obrigatório: não

GenericOutcome

Esse tipo de dados resulta de uma ação e produz um objeto com as seguintes propriedades:

Propriedades	Descrição
Bem-sucedida	Se a ação foi bem-sucedida ou não. Digite: bool Obrigatório: Sim
Erro	O erro que ocorreu se a ação não foi bem-sucedida. Digite: the section called “GameLiftError” Obrigatório: não

GenericOutcomeCallable

Esse tipo de dados é um resultado genérico assíncrono. Ele tem as seguintes propriedades:

Propriedades	Descrição
Bem-sucedida	Se a ação foi bem-sucedida ou não. Digite: bool Obrigatório: Sim

Propriedades	Descrição
Erro	O erro que ocorreu se a ação não foi bem-sucedida. Digite: the section called “GameLiftError” Obrigatório: não

PlayerSession

Esse tipo de dados representa uma sessão do jogador que Amazon GameLift Servers passa para o servidor do jogo. Para obter mais informações, consulte [PlayerSession](#).

Propriedades	Descrição
CreationTime	Digite: long Obrigatório: não
FleetId	Digite: std::string Obrigatório: não
GameSessionId	Digite: std::string Obrigatório: não
IpAddress	Digite: std::string Obrigatório: não
PlayerData	Digite: std::string Obrigatório: não
PlayerId	Digite: std::string Obrigatório: não
PlayerSessionId	Digite: std::string

Propriedades	Descrição
	Obrigatório: não
Porta	Digite: <code>int</code> Obrigatório: não
Status	Status da sessão de jogador para filtrar resultados. Quando um <code>PlayerSessionId</code> ou <code>PlayerId</code> é fornecido, o <code>PlayerSessionStatusFilter</code> tem efeito na resposta. Tipo: Um <code>PlayerSessionStatus</code> enum. Os valores possíveis incluem o seguinte: • <code>ACTIVE</code> • <code>COMPLETED</code> • <code>NOT_SET</code> • <code>RESERVED</code> • <code>TIMEDOUT</code> Obrigatório: não
TerminationTime	Digite: <code>long</code> Obrigatório: não
DnsName	Digite: <code>std::string</code> Obrigatório: não

StartMatchBackfillOutcome

Esse tipo de dados resulta de uma ação e produz um objeto com as seguintes propriedades:

Propriedades	Descrição
Resultado	O resultado da ação.

Propriedades	Descrição
	Digite: the section called “StartMatchBackfillResult” Obrigatório: não
ResultWithOwnership	O resultado da ação, convertido como um rvalue, para que o código de chamada possa se apropriar do objeto. Digite: <code>StartMatchBackfillResult&&</code> Obrigatório: não
Bem-sucedida	Se a ação foi bem-sucedida ou não. Digite: <code>bool</code> Obrigatório: Sim
Erro	O erro que ocorreu se a ação não foi bem-sucedida. Digite: the section called “GameLiftError” Obrigatório: não

StartMatchBackfillResult

Esse tipo de dados resulta de uma ação e produz um objeto com as seguintes propriedades:

Propriedades	Descrição
TicketId	Um identificador exclusivo para um tíquete de marcação de jogo. Se nenhum ID de tíquete for especificado aqui, Amazon GameLift Servers será gerado um na forma de um UUID. Use esse identificador para rastrear o status do ticket de alocação da partida e recuperar os resultados da partida. Digite: <code>std::string</code> Obrigatório: não

GetComputeCertificateOutcome

Esse tipo de dados resulta de uma ação e produz um objeto com as seguintes propriedades:

Propriedades	Descrição
Resultado	O resultado da ação. Digite: the section called “GetComputeCertificateResult” Obrigatório: não
ResultWithOwnership	O resultado da ação, convertido como um rvalue, para que o código de chamada possa se apropriar do objeto. Digite: <code>Aws::GameLift::Server::Model::GetComputeCertificateResult&&</code> Obrigatório: não
Bem-sucedida	Se a ação foi bem-sucedida ou não. Digite: <code>bool</code> Obrigatório: Sim
Erro	O erro que ocorreu se a ação não foi bem-sucedida. Digite: the section called “GameLiftError” Obrigatório: não

GetComputeCertificateResult

O caminho para o certificado TLS em seu computador e o nome do host do computador.

Propriedades	Descrição
CertificatePath	O caminho para o certificado TLS em seu recurso computacional. Ao usar uma frota Amazon GameLift Servers gerenciada, esse caminho contém:

Propriedades	Descrição
	<code>certificate.pem</code> : o certificado do usuário final. A cadeia completa de certificados é a combinação de <code>certificateChain.pem</code> anexados a esse certificado. <code>certificateChain.pem</code> : a cadeia de certificados que contém o certificado raiz e os certificados intermediários. <code>rootCertificate.pem</code> : o certificado raiz. <code>privateKey.pem</code> : a chave privada para o certificado do usuário final. Digite: <code>std::string</code> Obrigatório: não
ComputeName	O nome do seu recurso computacional. Digite: <code>std::string</code> Obrigatório: não

GetFleetRoleCredentialsOutcome

Esse tipo de dados resulta de uma ação e produz um objeto com as seguintes propriedades:

Propriedades	Descrição
Resultado	O resultado da ação. Digite: the section called “GetFleetRoleCredentialsResult” Obrigatório: não
ResultWithOwnership	O resultado da ação, convertido como um rvalue, para que o código de chamada possa se apropriar do objeto. Digite: <code>Aws::GameLift::Server::Model::GetFleetRoleCredentialsResult</code>

Propriedades	Descrição
	Obrigatório: não
Bem-sucedida	Se a ação foi bem-sucedida ou não. Digite: <code>bool</code> Obrigatório: Sim
Erro	O erro que ocorreu se a ação não foi bem-sucedida. Digite: the section called “GameLiftError” Obrigatório: não

GetFleetRoleCredentialsResult

Propriedades	Descrição
AccessKeyId	O ID da chave de acesso para autenticar e fornecer acesso aos seus recursos AWS . Digite: <code>string</code> Obrigatório: não
AssumedRoleId	O ID do usuário ao qual o perfil de serviço pertence. Digite: <code>string</code> Obrigatório: não
AssumedRoleUserArn	O nome do recurso da Amazon (ARN) do usuário ao qual pertence o perfil de serviço. Digite: <code>string</code> Obrigatório: não

Propriedades	Descrição
Expiração	A quantidade de tempo até que suas credenciais de sessão expirem. Digite: <code>DateTime</code> Obrigatório: não
SecretAccessKey	O ID da chave de acesso secreta para autenticação. Digite: <code>string</code> Obrigatório: não
SessionToken	Um token para identificar a sessão ativa atual interagindo com seus AWS recursos. Digite: <code>string</code> Obrigatório: não
Bem-sucedida	Se a ação foi bem-sucedida ou não. Digite: <code>bool</code> Obrigatório: Sim
Erro	O erro que ocorreu se a ação não foi bem-sucedida. Digite: the section called “GameLiftError” Obrigatório: não

Inicializar SDKOutcome

Note

InitSDKOutcome é retornado somente quando você cria o SDK com a bandeira std. Se você construir com a nostd bandeira, ela será [the section called “GenericOutcome”](#) retornada em vez disso.

Propriedades	Descrição
Bem-sucedida	Se a ação foi bem-sucedida ou não. Digite: bool Obrigatório: Sim
Erro	O erro que ocorreu se a ação não foi bem-sucedida. Digite: the section called “GameLiftError” Obrigatório: não

GameLiftError

Propriedades	Descrição
ErrorType	O tipo de erro. Tipo: Um GameLiftErrorType enum . Obrigatório: não
ErrorName	O nome do erro. Digite: std::string Obrigatório: não
ErrorMessage	A mensagem de erro.

Propriedades	Descrição
	Digite: <code>std::string</code>
	Obrigatório: não

Enumerações

Os enums definidos para o SDK do servidor para Amazon GameLift Servers (C++) são definidos da seguinte forma:

GameLiftErrorType

Valor da string indicando o tipo de erro. Os valores válidos são:

- `BAD_REQUEST_EXCEPTION`
- `GAMESESSION_ID_NOT_SET` – O ID da sessão do jogo não foi definido.
- `INTERNAL_SERVICE_EXCEPTION`
- `LOCAL_CONNECTION_FAILED` — A conexão local falhou. Amazon GameLift Servers
- `NETWORK_NOT_INITIALIZED` – A rede não foi inicializada.
- `SERVICE_CALL_FAILED` — Uma chamada para um serviço falhou. AWS
- `WEBSOCKET_CONNECT_FAILURE`
- `WEBSOCKET_CONNECT_FAILURE_FORBIDDEN`
- `WEBSOCKET_CONNECT_FAILURE_INVALID_URL`
- `WEBSOCKET_CONNECT_FAILURE_TIMEOUT`
- `ALREADY_INITIALIZED` — O Amazon GameLift Servers servidor ou cliente já foi inicializado com `Initialize ()`.
- `FLEET_MISMATCH` – A frota de destino não corresponde à frota de uma `GameSession` ou `PlayerSession`.
- `GAMELIFT_CLIENT_NOT_INITIALIZED` — O cliente não foi inicializado. Amazon GameLift Servers
- `GAMELIFT_SERVER_NOT_INITIALIZED` — O servidor não foi inicializado. Amazon GameLift Servers
- `GAME_SESSION_ENDED_FAILED` — o SDK do servidor não conseguiu entrar em contato com o serviço para informar Amazon GameLift Servers que a sessão do jogo foi encerrada.

- `GAME_SESSION_NOT_READY` — A sessão de jogo do servidor não foi ativadaAmazon GameLift Servers.
- `GAME_SESSION_READY_FAILED` — o SDK do servidor não conseguiu entrar em contato com o serviço para informar Amazon GameLift Servers que a sessão do jogo está pronta.
- `INITIALIZATION_MISMATCH` – Um método de cliente foi chamado depois de `Server::Initialize()`, ou vice-versa.
- `NOT_INITIALIZED` — O Amazon GameLift Servers servidor ou cliente não foi inicializado com `Initialize ()`.
- `NO_TARGET_ALIASID_SET` – Um aliasID de destino não foi definido.
- `NO_TARGET_FLEET_SET` – Uma frota alvo não foi definida.
- `PROCESS_ENDING_FAILED` — o SDK do servidor não conseguiu entrar em contato com o serviço para Amazon GameLift Servers informar que o processo está sendo encerrado.
- `PROCESS_NOT_ACTIVE` — O processo do servidor ainda não está ativo, não está vinculado a um e não pode `GameSession` aceitar ou processar. `PlayerSessions`
- `PROCESS_NOT_READY` – O processo do servidor ainda não está pronto para ser ativado.
- `PROCESS_READY_FAILED` — o SDK do servidor não Amazon GameLift Servers conseguiu entrar em contato com o serviço para informar que o processo está pronto.
- `SDK_VERSION_DETECTION_FAILED` – Falha na detecção da versão do SDK.
- `STX_CALL_FAILED` — Uma chamada para o componente de back-end do XStx servidor falhou.
- `STX_INITIALIZATION_FAILED` — O componente de back-end do XStx servidor falhou ao inicializar.
- `UNEXPECTED_PLAYER_SESSION` – Uma sessão de jogador não registrada foi encontrada pelo servidor.
- `WEBSOCKET_CONNECT_FAILURE`
- `WEBSOCKET_CONNECT_FAILURE_FORBIDDEN`
- `WEBSOCKET_CONNECT_FAILURE_INVALID_URL`
- `WEBSOCKET_CONNECT_FAILURE_TIMEOUT`
- `WEBSOCKET_RETRIABLE_SEND_MESSAGE_FAILURE` — Falha recuperável ao enviar uma mensagem ao Serviço. GameLift WebSocket
- `WEBSOCKET_SEND_MESSAGE_FAILURE` — Falha ao enviar uma mensagem para o serviço. GameLift WebSocket

- MATCH_BACKFILL_REQUEST_VALIDATION – A validação da solicitação falhou.
- PLAYER_SESSION_REQUEST_VALIDATION – A validação da solicitação falhou.

PlayerSessionCreationPolicy

Valor de string que indica se a sessão do jogo aceita novos jogadores. Os valores válidos são:

- ACCEPT_ALL – Aceite todas as novas sessões de jogador.
- DENY_ALL – Recuse todas as novas sessões de jogador.
- NOT_SET – A sessão do jogo não está configurada para aceitar ou negar sessões de novos jogadores.

SDK 5.x do servidor C++ para Amazon GameLift Servers -- Tipos de dados

Tópicos

- [GetSdkVersion\(\)](#)
- [InitSDK\(\)](#)
- [InitSDK\(\)](#)
- [ProcessReady\(\)](#)
- [ProcessReadyAsync\(\)](#)
- [ProcessEnding\(\)](#)
- [ActivateGameSession\(\)](#)
- [UpdatePlayerSessionCreationPolicy\(\)](#)
- [GetGameSessionId\(\)](#)
- [GetTerminationTime\(\)](#)
- [AcceptPlayerSession\(\)](#)
- [RemovePlayerSession\(\)](#)
- [DescribePlayerSessions\(\)](#)
- [StartMatchBackfill\(\)](#)
- [StopMatchBackfill\(\)](#)
- [GetComputeCertificate\(\)](#)
- [GetFleetRoleCredentials\(\)](#)
- [Destroy\(\)](#)

GetSdkVersion()

Retorna o número da versão atual do SDK compilado no processo de servidor.

Sintaxe

```
Aws::GameLift::AwsStringOutcome Server::GetSdkVersion();
```

Valor de retorno

Se bem-sucedido, retornará a versão do SDK atual como um objeto [the section called “AwsStringOutcome”](#). O objeto retornada inclui apenas o número da versão (exemplo, 5.0.0). Se não for bem-sucedido, retornará uma mensagem de erro.

Exemplo

```
Aws::GameLift::AwsStringOutcome SdkVersionOutcome =  
    Aws::GameLift::Server::GetSdkVersion();
```

InitSDK()

Inicializa o Amazon GameLift Servers SDK. Chame esse método na inicialização, antes de qualquer outra etapa de inicialização relacionada a. Amazon GameLift Servers Essa ação lê os parâmetros do servidor do ambiente host para configurar a comunicação entre o processo do servidor do jogo e o Amazon GameLift Servers serviço.

Se a versão do servidor do jogo for implantada sem a frota ou frota de contêineres do Amazon GameLift Servers Agent to a Amazon GameLift Servers Anywhere, ligue [InitSDK\(\)](#) e especifique um conjunto de parâmetros do servidor.

Sintaxe

```
Server::InitSDKOutcome Server::initSdkOutcome = InitSDK();
```

Valor de retorno

Retorna um objeto [the section called “Inicializar SDKOutcome”](#) que indica se o processo do servidor está pronto para ser chamado [ProcessReady\(\)](#).

Exemplo

```
//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()

Inicializa o Amazon GameLift Servers SDK. Chame esse método na inicialização, antes de qualquer outra etapa de inicialização relacionada a. Amazon GameLift Servers Essa ação requer um conjunto de parâmetros do servidor para configurar a comunicação entre o processo do servidor do jogo e o Amazon GameLift Servers serviço.

Se a versão do servidor do jogo for implantada em uma EC2 frota Amazon GameLift Servers gerenciada ou Amazon GameLift Servers em uma frota ou frota de contêineres Anywhere com o Amazon GameLift Servers Agente, ligue [InitSDK\(\)](#) sem parâmetros do servidor.

Sintaxe

```
Server::InitSDKOutcome Server::initSdkOutcome = InitSDK(serverParameters);
```

Parâmetros

[ServerParameters](#)

Para inicializar um servidor de jogo em uma frota Amazon GameLift Servers Anywhere, construa um `ServerParameters` objeto com as seguintes informações:

- O URL do WebSocket usado para se conectar ao seu servidor de jogo.
- O ID do processo usado para hospedar o servidor de jogos.
- O ID do computador que hospeda os processos do seu servidor de jogos.
- O ID da Amazon GameLift Servers frota que contém sua computação Amazon GameLift Servers Anywhere.
- O token de autorização gerado pela Amazon GameLift Servers operação.

Valor de retorno

Retorna um objeto [the section called “Inicializar SDKOutcome”](#) que indica se o processo do servidor está pronto para ser chamado [ProcessReady\(\)](#).

Note

Se as chamadas para `InitSDK()` estiverem falhando para compilações de jogos implantadas em frotas do Anywhere, verifique o parâmetro `ServerSdkVersion` usado ao criar o recurso de compilação. Você deve definir explicitamente esse valor para a versão do SDK do servidor em uso. O valor padrão desse parâmetro é 4.x, o que não é compatível. Para resolver esse problema, crie uma nova versão e implante-a em uma nova frota.

Exemplo

Amazon GameLift ServersExemplo em qualquer lugar

```
//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 que o processo do servidor está pronto para hospedar sessões de jogo. Chame esse método após invocar [InitSDK\(\)](#). Esse método deve ser chamado somente uma vez por processo.

Sintaxe

```
GenericOutcome ProcessReady(const Aws::GameLift::Server::ProcessParameters
&processParameters);
```

Parâmetros

processParameters

Um objeto [ProcessParameters](#) que fornece as seguintes informações sobre o processo do servidor:

- Nomes dos métodos de retorno de chamada implementados no código do servidor do jogo que o Amazon GameLift Servers serviço invoca para se comunicar com o processo do servidor.
- Número da porta em que o processo de servidor está escutando.
- Caminho de qualquer arquivo específico da sessão do jogo que você deseja que o Amazon GameLift Servers capture e armazene.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

Este exemplo ilustra as implementações das funções de chamada e delegação [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 o serviço Amazon GameLift Servers de que o processo do servidor está pronto para hospedar sessões do jogo. Este método deve ser chamado depois que o processo do servidor estiver pronto para hospedar uma sessão de jogo. Os parâmetros especificam os nomes das funções de retorno de chamada Amazon GameLift Servers a serem chamadas em determinadas circunstâncias. O código do servidor de jogos deve implementar essas funções.

Essa chamada é assíncrona. Para fazer uma chamada síncrona, use [ProcessReady\(\)](#). Consulte [Inicializar o processo do servidor](#) para obter mais detalhes.

Sintaxe

```
GenericOutcomeCallable ProcessReadyAsync(
    const Aws::GameLift::Server::ProcessParameters &processParameters);
```

Parâmetros

processParameters

Um objeto [ProcessParameters](#) que fornece as seguintes informações sobre o processo do servidor:

- Nomes dos métodos de retorno de chamada implementados no código do servidor do jogo que o Amazon GameLift Servers serviço invoca para se comunicar com o processo do servidor.
- Número da porta em que o processo de servidor está escutando.
- Caminho de qualquer arquivo específico da sessão do jogo que você deseja que o Amazon GameLift Servers capture e armazene.

Obrigatório: Sim

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

```
// 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 que o processo do servidor está sendo encerrado. Chame esse método depois de todas as outras tarefas de limpeza (incluindo o encerramento da sessão ativa do jogo) e antes de encerrar o processo. Dependendo do resultado `ProcessEnding()`, o processo sai com sucesso (0) ou erro (-1) e gera um evento de frota. Se o processo for encerrado com um erro, o evento de frota gerado será `SERVER_PROCESS_TERMINATED_UNHEALTHY`.

Sintaxe

```
Aws::GameLift::GenericOutcome processEndingOutcome =
    Aws::GameLift::Server::ProcessEnding();
```

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

Este exemplo chama `ProcessEnding()` e `Destroy()` antes de encerrar o processo do servidor com um código de saída bem-sucedido ou de erro.

```
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 que o processo do servidor ativou uma sessão de jogo e agora está pronto para receber conexões de jogadores. Essa ação deve ser chamada como parte da função de retorno de chamada `onStartGameSession()`, após toda inicialização da sessão do jogo.

Sintaxe

```
Aws::GameLift::GenericOutcome activateGameSessionOutcome =
    Aws::GameLift::Server::ActivateGameSession();
```

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

Este exemplo mostra chamado `ActivateGameSession()` como parte da função de delegação `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();
}
```

UpdatePlayerSessionCreationPolicy()

Atualiza a capacidade da sessão do jogo atual para aceitar novas sessões de jogador. Atualiza a capacidade da sessão do jogo atual para aceitar novas sessões de jogador.

Sintaxe

```
GenericOutcome  
UpdatePlayerSessionCreationPolicy(Aws::GameLift::Model::PlayerSessionCreationPolicy  
newPlayerSessionPolicy);
```

Parâmetros

playerCreationSessionPolítica

Tipo: `PlayerSessionCreationPolicy` valor [enum](#).

Obrigatório: Sim

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

Este exemplo define a política de ingresso da sessão do jogo atual para aceitar todos os jogadores.

```
Aws::GameLift::GenericOutcome outcome =  
  
    Aws::GameLift::Server::UpdatePlayerSessionCreationPolicy(Aws::GameLift::Model::PlayerSessionCr
```

GetGameSessionId()

Recupera o ID da sessão de jogo hospedada pelo processo do servidor ativo.

Para processos inativos que não são ativados com uma sessão de jogo, a chamada retorna um [the section called “GameLiftError”](#).

Sintaxe

```
AwsStringOutcome GetGameSessionId()
```

Parâmetros

Essa ação não tem um parâmetro.

Valor de retorno

Se bem-sucedido, retornará o ID da sessão de jogo como um objeto [the section called "AwsStringOutcome"](#). Se não for bem-sucedido, retornará uma mensagem de erro.

Para processos inativos que não são ativados com uma sessão de jogo, a chamada retorna `Success=True` e `GameSessionId=""`.

Exemplo

```
Aws::GameLift::AwsStringOutcome sessionIdOutcome =  
    Aws::GameLift::Server::GetGameSessionId();
```

GetTerminationTime()

Retorna a hora em que um processo do servidor está programado para ser desligado, se essa informação estiver disponível. Um processo do servidor age após receber um `onProcessTerminate()` retorno de chamada de Amazon GameLift Servers. Amazon GameLift Servers chamadas `onProcessTerminate()` pelos seguintes motivos:

- Quando o processo do servidor relatou problemas de saúde ou não respondeu Amazon GameLift Servers.
- Ao encerrar a instância durante um evento de redução.
- Quando uma instância é encerrada devido a uma [interrupção na instância spot](#).

Sintaxe

```
AwsDateTimeOutcome GetTerminationTime()
```

Valor de retorno

Se for bem-sucedido, retornará o horário de término como um objeto `AwsDateTimeOutcome`. O valor é o tempo de rescisão, expresso em tiques decorridos desde então `0001 00:00:00`. Por exemplo, o valor da data e hora `2020-09-13 12:26:40 -000Z` é igual aos tiques `637355968000000000`. Se nenhum horário de encerramento estiver disponível, o retornará uma mensagem de erro.

Se o processo não tiver recebido um `ProcessParameters.OnProcessTerminate()` callback, uma mensagem de erro é retornada. Para obter mais informações sobre como desligar um processo do servidor, consulte [Responder a uma notificação de desligamento do processo do servidor](#).

Exemplo

```
Aws::GameLift::AwsLongOutcome TermTimeOutcome =  
    Aws::GameLift::Server::GetTerminationTime();
```

AcceptPlayerSession()

Notifica Amazon GameLift Servers que um jogador com o ID de sessão de jogador especificado se conectou ao processo do servidor e precisa de validação. Amazon GameLift Servers verifica se o ID da sessão do jogador é válido. Depois que a sessão do jogador for validada, o status da vaga do jogador é Amazon GameLift Servers alterado de RESERVADO para ATIVO.

Sintaxe

```
GenericOutcome AcceptPlayerSession(String playerId)
```

Parâmetros

playerSessionId

ID exclusivo emitido Amazon GameLift Servers quando uma nova sessão de jogador é criada.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

Este exemplo trata de uma solicitação de conexão que inclui a validação e a rejeição de uma sessão de jogador inválida. IDs

```
void ReceiveConnectingPlayerSessionID (Connection& connection, const std::string&  
    playerId)  
{  
    Aws::GameLift::GenericOutcome connectOutcome =  
        Aws::GameLift::Server::AcceptPlayerSession(playerSessionId);  
    if(connectOutcome.IsSuccess())
```

```
{
    connectionToSessionMap.emplace(connection, playerSessionId);
    connection.Accept();
}
else
{
    connection.Reject(connectOutcome.GetError().GetMessage());
}
}
```

RemovePlayerSession()

Notifica Amazon GameLift Servers que um jogador se desconectou do processo do servidor. Em resposta, Amazon GameLift Servers altera o slot do jogador para disponível.

Sintaxe

```
GenericOutcome RemovePlayerSession(String playerSessionId)
```

Parâmetros

playerSessionId

ID exclusivo emitido Amazon GameLift Servers quando uma nova sessão de jogador é criada.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

```
Aws::GameLift::GenericOutcome disconnectOutcome =
    Aws::GameLift::Server::RemovePlayerSession(playerSessionId);
```

DescribePlayerSessions()

Recupera dados da sessão do jogador, que incluem configurações, metadados da sessão e dados do jogador. Use esse método para obter informações sobre o seguinte:

- Uma sessão para um jogador
- Todas as sessões de jogador em uma sessão de jogo

- Todas as sessões de jogadores associadas a um único ID de jogador

Sintaxe

```
DescribePlayerSessionsOutcome DescribePlayerSessions(DescribePlayerSessionsRequest  
describePlayerSessionsRequest)
```

Parâmetros

[DescribePlayerSessionsRequest](#)

Um objeto [the section called “DescribePlayerSessionsRequest”](#) que descreve quais sessões de jogador recuperar.

Valor de retorno

Se bem-sucedido, retorna um objeto [the section called “DescribePlayerSessionsOutcome”](#) que contém um conjunto de objetos de sessão do jogador que atendem aos parâmetros de solicitação.

Exemplo

Este exemplo solicita todas as sessões de jogador conectadas ativamente a uma sessão de jogo especificada. Ao omitir NextToken e definir o valor limite como 10, Amazon GameLift Servers retorna os primeiros registros de sessão de 10 jogadores que correspondam à solicitação.

```
// 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()

Envia uma solicitação para encontrar novos jogadores para os slots abertos em uma sessão de jogo criada com o FlexMatch. Para obter mais informações, consulte o [recurso FlexMatch de preenchimento](#).

Esta ação é assíncrona. Se novos jogadores forem combinados, Amazon GameLift Servers fornece dados atualizados do matchmaker usando a função de retorno de chamada.

`OnUpdateGameSession()`

Um processo de servidor pode ter apenas uma solicitação de alocação de correspondência ativa por vez. Para enviar uma nova solicitação, primeiro chame [StopMatchBackfill\(\)](#) para cancelar a solicitação original.

Sintaxe

```
StartMatchBackfillOutcome StartMatchBackfill (StartMatchBackfillRequest  
startBackfillRequest);
```

Parâmetros

[StartMatchBackfillRequest](#)

Um `StartMatchBackfillRequest` objeto que comunica as seguintes informações:

- ID do tíquete a ser atribuído à solicitação de alocação. Essas informações são opcionais; se nenhum ID for fornecido, Amazon GameLift Servers será gerado um.
- O marcador de jogos para o qual a solicitação é enviada. O ARN completo da configuração é necessário. Esse valor está nos dados do marcador de jogos da sessão de jogo.
- O ID da sessão de jogo a ser preenchida.
- Os dados disponíveis de marcação para os jogadores atuais da sessão do jogo.

Valor de retorno

Retorna um objeto [the section called “StartMatchBackfillOutcome”](#) com o ID do tíquete de alocação de correspondência ou um falha com uma mensagem de erro.

Exemplo

```
// 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()

Cancela uma solicitação de alocação de correspondência ativa. Para obter mais informações, consulte o [recurso FlexMatch de preenchimento](#).

Sintaxe

```
GenericOutcome StopMatchBackfill (StopMatchBackfillRequest stopBackfillRequest);
```

Parâmetros

StopMatchBackfillRequest

Um StopMatchBackfillRequest objeto identificando o tíquete de matchmaking a ser cancelado:

- O ID do ticket atribuído à solicitação de alocação.
- O marcador de jogo que recebeu a solicitação de alocação.
- A sessão do jogo associada à solicitação de alocação.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

```
// 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 o caminho para o certificado TLS usado para criptografar a conexão de rede entre seu recurso computacional Amazon GameLift Servers Anywhere e Amazon GameLift Servers. Você pode usar o caminho do certificado ao registrar seu dispositivo computacional Amazon GameLift Servers em uma frota Anywhere. Para obter mais informações, consulte [RegisterCompute](#).

Sintaxe

```
GetComputeCertificateOutcome Server::GetComputeCertificate()
```

Valor de retorno

Retorna um [the section called “GetComputeCertificateOutcome”](#).

Exemplo

```
Aws::GameLift::GetComputeCertificateOutcome certificate =
    Aws::GameLift::Server::GetComputeCertificate();
```

GetFleetRoleCredentials()

Recupera as credenciais da função do IAM que autorizam Amazon GameLift Servers a interação com outras pessoas. Serviços da AWS Para obter mais informações, consulte [Comunique-se com outros AWS recursos de suas frotas](#).

Sintaxe

```
GetFleetRoleCredentialsOutcome GetFleetRoleCredentials(GetFleetRoleCredentialsRequest request);
```

Parâmetros

[GetFleetRoleCredentialsRequest](#)

Valor de retorno

Informa um objeto [the section called “GetFleetRoleCredentialsOutcome”](#).

Exemplo

```
// 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);
```

Este exemplo mostra o uso do valor `RoleSessionName` opcional para atribuir um nome à sessão de credenciais para fins de auditoria. Se você não fornecer um nome de sessão de função, o valor padrão "*[fleet-id]-[host-id]*" será usado.

```
// 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);
```

Destroy()

Libera o SDK Amazon GameLift Servers do servidor de jogos da memória. Como melhor prática, chame esse método antes `ProcessEnding()` e depois de encerrar o processo. Se você estiver

usando uma frota Anywhere e não estiver encerrando os processos do servidor após cada sessão de jogo, ligue `Destroy()` e reinicialize antes de notificar Amazon GameLift Servers que o processo está pronto para hospedar uma sessão de jogo. `InitSDK()` `ProcessReady()`

Sintaxe

```
GenericOutcome Aws::GameLift::Server::Destroy();
```

Parâmetros

Não há parâmetros.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

```
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);  
}
```

SDK 5.x do servidor C# para Amazon GameLift Servers -- Ações

Use a referência 5.x do SDK do servidor para integrar seu jogo multijogador para hospedagem com Amazon GameLift Servers Para obter orientação sobre o processo de integração, consulte [Adicione Amazon GameLift Servers ao seu servidor de jogos](#). Se você estiver usando o Amazon GameLift Servers plug-in para Unity, veja também [Amazon GameLift Serversplugin para Unity \(servidor SDK 5.x\)](#).

SDK 5.x do servidor C# para Amazon GameLift Servers -- Tipos de dados

Use a referência 5.x do SDK do servidor Amazon GameLift Servers C# para integrar seu jogo multijogador para hospedagem com. Amazon GameLift Servers Para obter orientação sobre o processo de integração, consulte [Adicione Amazon GameLift Servers ao seu servidor de jogos](#). Se você estiver usando o Amazon GameLift Servers plug-in para Unity, veja também [Amazon GameLift Serversplugin para Unity \(servidor SDK 5.x\)](#).

[SDK 5.x do servidor C# para Amazon GameLift Servers -- Ações](#)

Tipos de dados

- [LogParameters](#)
- [ProcessParameters](#)
- [UpdateGameSession](#)
- [GameSession](#)
- [ServerParameters](#)
- [StartMatchBackfillRequest](#)
- [Jogador](#)
- [DescribePlayerSessionsRequest](#)
- [StopMatchBackfillRequest](#)
- [GetFleetRoleCredentialsRequest](#)
- [AttributeValue](#)
- [AwsStringOutcome](#)
- [GenericOutcome](#)
- [DescribePlayerSessionsOutcome](#)
- [DescribePlayerSessionsResult](#)
- [PlayerSession](#)
- [StartMatchBackfillOutcome](#)
- [StartMatchBackfillResult](#)
- [GetComputeCertificateOutcome](#)
- [GetComputeCertificateResult](#)
- [GetFleetRoleCredentialsOutcome](#)
- [GetFleetRoleCredentialsResult](#)

- [AwsDateTimeOutcome](#)
- [GameLiftError](#)
- [Enumerações](#)

LogParameters

Use esse tipo de dados para identificar quais arquivos gerados durante uma sessão de jogo você deseja que o servidor do jogo envie Amazon GameLift Servers após o término da sessão de jogo. O servidor do jogo se comunica LogParameters to Amazon GameLift Servers em uma [ProcessReady\(\)](#) chamada.

Properties	Descrição
LogPaths	A lista de caminhos de diretório para os arquivos de log do servidor de jogos que você Amazon GameLift Servers deseja armazenar para acesso futuro. O processo do servidor gera esses arquivos durante cada sessão de jogo. Você define caminhos e nomes de arquivos em seu servidor de jogo e os armazena no diretório raiz de construção do jogo. Os caminhos de log devem ser absolutos. Por exemplo, se a compilação do jogo armazenar logs de sessão de jogo em um caminho como MyGame\sessionLogs\ , o caminho estará c:\game\MyGame\sessionLogs em uma instância do Windows. Digite: List<String> Obrigatório: não

ProcessParameters

Esse tipo de dados contém o conjunto de parâmetros enviados Amazon GameLift Servers em uma [ProcessReady\(\)](#) chamada.

Properties	Descrição
LogParameters	O objeto com uma lista de caminhos de diretório para arquivos de log da sessão de jogo. Digite: <code>Aws::GameLift::Server:: LogParameters</code> Obrigatório: Sim
OnHealthCheck	O nome da função de retorno de chamada que Amazon GameLift Servers invoca para solicitar um relatório de status de saúde do processo do servidor. Amazon GameLift Servers chama essa função a cada 60 segundos. Depois de chamar essa função, Amazon GameLift Servers aguarda 60 segundos por uma resposta; se nenhuma for recebida, Amazon GameLift Servers registra o processo do servidor como não íntegro. Digite: <code>void OnHealthCheckDelegate()</code> Obrigatório: Sim
OnProcessTerminate	O nome da função de retorno de chamada que é Amazon GameLift Servers invocada para forçar o encerramento do processo do servidor. Depois de chamar essa função, o Amazon GameLift Servers aguardará cinco minutos para o processo de servidor desligar e responder com uma chamada ProcessEn

	ding() antes do desligamento do processo de servidor. Digite: <code>void OnProcessTerminate Delegate()</code> Obrigatório: Sim
OnStartGameSession	O nome da função de retorno de chamada que é Amazon GameLift Servers invocada para ativar uma nova sessão de jogo. Amazon GameLift Servers chama essa função em resposta à solicitação do cliente CreateGameSession. A função de retorno de chamada recebe um objeto GameSession. Digite: <code>void OnStartGameSession Delegate(GameSession)</code> Obrigatório: Sim
OnUpdateGameSession	O nome da função de retorno de chamada que Amazon GameLift Servers invoca para passar um objeto de sessão de jogo atualizado para o processo do servidor. Amazon GameLift Servers chama essa função quando uma solicitação de preenchimento de correspondência é processada para fornecer dados atualizados do matchmaker. Ele transmite um objeto GameSession, uma atualização de status (<code>updateReason</code>) e o ID do tíquete de alocação de correspondência. Tipo: vazio <code>OnUpdateGameSessionDelegate (UpdateGameSession)</code> Obrigatório: não

Porta

O número da porta em que o processo de servidor escuta para novas conexões de jogador. O valor deve estar dentro do intervalo de portas configurado para qualquer frota que implanta essa compilação do servidor de jogo. Esse número de porta está incluído nos objetos das sessões de jogo e jogador, que as sessões de jogo usam ao se conectar a um processo de servidor.

Digite: Integer

Obrigatório: Sim

UpdateGameSession

As informações atualizadas de um objeto de sessão de jogo incluem o motivo pelo qual a sessão de jogo foi atualizada. Se a atualização estiver relacionada a uma ação de alocação de correspondência, esse tipo de dados inclui o ID do tíquete de alocação.

Propriedades	Descrição
GameSession	Um objeto GameSession. O objeto <code>GameSession</code> contém propriedades que descrevem uma sessão de jogo. Digite: <code>GameSession GameSession()</code> Obrigatório: Sim
UpdateReason	O motivo pelo qual a sessão do jogo está sendo atualizada. Digite: <code>UpdateReason UpdateReason()</code> Obrigatório: Sim
BackfillTicketId	O ID do tíquete de alocação que está tentando atualizar a sessão do jogo.

Propriedades	Descrição
	Digite: String
	Obrigatório: Sim

GameSession

Detalhes de uma sessão de jogo.

Propriedades	Descrição
GameSessionId	Um identificador exclusivo da sessão de jogo. Um ARN de sessão de jogo tem o seguinte formato: <code>arn:aws:gamelift:<region>::gamesession/<fleet ID>/<custom ID string or idempotency token></code> . Digite: String Obrigatório: não
Name	Um rótulo descritivo da sessão do jogo. Digite: String Obrigatório: não
FleetId	Um identificador exclusivo da frota em que a sessão de jogo está sendo executada. Digite: String Obrigatório: não
MaximumPlayerSessionCount	O número máximo de conexões do jogador com a sessão do jogo. Digite: Integer

Propriedades	Descrição
	Obrigatório: não
Porta	O número da porta da sessão de jogo. Para se conectar a um servidor de Amazon GameLift Servers jogos, um aplicativo precisa do endereço IP e do número da porta. Digite: Integer Obrigatório: não
IpAddress	O endereço IP da sessão de jogo. Para se conectar a um servidor de Amazon GameLift Servers jogos, um aplicativo precisa do endereço IP e do número da porta. Digite: String Obrigatório: não
GameSessionData	Um conjunto de propriedades de sessão de jogo personalizadas, formatadas como um único valor de string. Digite: String Obrigatório: não

Propriedades	Descrição
MatchmakerData	As informações sobre o processo de marcação de jogos usado para criar a sessão do jogo, na sintaxe JSON, formatado como uma string. Além da configuração de marcação de jogos usada, ela contém dados sobre todos os jogadores designados para a partida, incluindo atributos do jogador e atribuições da equipe. Digite: <code>String</code> Obrigatório: não
GameProperties	Um conjunto de propriedades personalizadas para uma sessão de jogo, formatado como pares de chave:valor. Essas propriedades são passadas com uma solicitação de início de uma nova sessão de jogo. Digite: <code>Dictionary<string, string></code> Obrigatório: não

Propriedades	Descrição
DnsName	O identificador DNS atribuído à instância que está executando a sessão do jogo. Os valores têm o formato a seguir: • Frotas habilitadas para TLS: <unique identifier>.<region identifier>.amazongamelift.com . • Non-TLS-enabled frotas: ec2-<unique identifier>.compute.amazonaws.com . Ao se conectar a uma sessão de jogo que está sendo executada em uma frota compatível com TLS, você deve usar o nome DNS, não o endereço IP. Digite: String Obrigatório: não

ServerParameters

Informações usadas para manter a conexão entre um servidor Amazon GameLift Servers Anywhere e o Amazon GameLift Servers serviço. Essas informações são usadas ao iniciar novos processos de servidor com [InitSDK\(\)](#). Para servidores hospedados em EC2 instâncias Amazon GameLift Servers gerenciadas, use um objeto vazio.

Propriedades	Descrição
WebSocketUrl	O <code>GameLiftServerSdkEndpoint</code> devolvido quando você <code>RegisterCompute</code> fazia parte do Amazon GameLift Servers Anywhere. Digite: String

Propriedades	Descrição
	Obrigatório: Sim
ProcessId	Um identificador exclusivo registrado no processo do servidor que hospeda seu jogo. Digite: <code>String</code> Obrigatório: Sim
HostId	Um identificador exclusivo para o host com os processos do servidor que hospedam seu jogo. O HostID é o ComputeName usado quando você registrou seu computador. Para obter mais informações, consulte RegisterCompute Digite: <code>String</code> Obrigatório: Sim
FleetId	O ID da frota na qual o computador está registrado. Para obter mais informações, consulte, RegisterCompute. Digite: <code>String</code> Obrigatório: Sim
AuthToken	O token de autenticação gerado por Amazon GameLift Servers isso autentica seu servidor em Amazon GameLift Servers. Para obter mais informações, consulte, GetComputeAuthToken. Digite: <code>String</code> Obrigatório: Sim

StartMatchBackfillRequest

Informações usadas para criar uma solicitação de alocação de marcação de jogos. O servidor do jogo comunica essas informações Amazon GameLift Servers em uma [StartMatchBackfill\(\)](#) chamada.

Propriedades	Descrição
GameSessionArn	O identificador da sessão de jogo exclusivo . A operação de API retorna GetGameSessionId o identificador no formato de ARN. Digite: String Obrigatório: Sim
MatchmakingConfigurationArn	O identificador exclusivo, no formato de ARN, para o marcador de jogos a ser usado para essa solicitação. O marcador de jogos ARN da sessão de jogo original está no objeto de sessão de jogo na propriedade de dados do marcador de jogos. Saiba mais sobre os dados do marcador de jogos em Trabalhar com dados do marcador de jogos. Digite: String Obrigatório: Sim
Jogadores	Um conjunto de dados que representa todos os jogadores que estão na sessão do jogo no momento. O marcador de jogos usa essas informações para pesquisar novos jogadores que sejam correspondências ideais para os jogadores atuais. Digite: List<Player> Obrigatório: Sim

Propriedades	Descrição
TicketId	O identificador exclusivo de um tíquete de solicitação de marcação de jogo ou de alocação de correspondência. Se você não fornecer um valor, Amazon GameLift Servers gera um. Use esse identificador para acompanhar o status do tíquete de alocação de correspondência ou cancelar a solicitação, se necessário. Digite: <code>String</code> Obrigatório: não

Jogador

Representa um jogador em marcação de jogos. Quando uma solicitação de matchmaking começa, um jogador tem um ID de jogador, atributos, e possivelmente dados de latência. Amazon GameLift Servers adiciona informações da equipe após a realização de uma partida.

Propriedades	Descrição
LatencyInMS	Um conjunto de valores expressos em milissegundos, que indicam a quantidade de latência que um jogador experimenta quando conectado a um local. Se essa propriedade for usada, o player só será correspondido aos locais listados. Se um marcador de jogos tiver uma regra que avalie a latência do jogador, os jogadores deverão reportar a latência a ser correspondida. Digite: <code>Dictionary<string, int></code> Obrigatório: não

Propriedades	Descrição
PlayerAttributes	Uma coleção de pares de chave:valor que contêm informações de jogadores para uso em marcação de jogos. As chaves de atributos do jogador devem corresponder às PlayerAttributes usadas em um conjunto de regras de matchmaking. Para obter mais informações sobre os atributos do jogador, consulte AttributeValue. Digite: Dictionary<string, Attribute Value> Obrigatório: não
PlayerId	Um identificador exclusivo para um jogador. Digite: String Obrigatório: não
Equipe	O nome da equipe à qual o jogador está designado em uma partida. Você define o nome da equipe no conjunto de regras de marcação de jogos. Digite: String Obrigatório: não

DescribePlayerSessionsRequest

Esse tipo de dados é usado para especificar quais sessões de jogador recuperar. Ele pode ser usado de várias maneiras: (1) forneça PlayerSessionId a para solicitar uma sessão específica do jogador; (2) forneça GameSessionId a para solicitar todas as sessões do jogador na sessão de jogo especificada; ou (3) forneça PlayerId a para solicitar todas as sessões do jogador especificado. Para

coleções maiores de sessões de jogador, use os parâmetros de paginação para recuperar resultados como páginas sequenciais.

Propriedades	Descrição
GameSessionId	O identificador da sessão de jogo exclusivo . Use esse parâmetro para solicitar todas as sessões de jogador da sessão de jogo especificada. O formato do ID da sessão de jogo é o seguinte: <code>arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string></code> . O valor de <code><ID string></code> é uma string de ID personalizada (se uma tiver sido especificada quando a sessão do jogo foi criada) ou uma string gerada. Digite: String Obrigatório: não
PlayerSessionId	O Identificador exclusivo para uma sessão de jogador. Digite: String Obrigatório: não
PlayerId	Um identificador exclusivo para um jogador. Consulte Gerar jogador IDs. Digite: String Obrigatório: não
PlayerSessionStatusFilter	O status da sessão de jogador para filtrar resultados. Entre os status da sessão de jogador possíveis estão os seguintes: • RESERVADO — A solicitação de sessão do jogador foi recebida, mas o jogador ainda

Propriedades	Descrição
	não se conectou ao servidor. O processo and/or foi validado. • ACTIVE – O jogador foi validado pelo processo de servidor e está conectado no momento. • COMPLETED – A conexão do jogador foi interrompida. • TEMPO LIMITE — Uma solicitação de sessão do jogador foi recebida, mas o jogador que não se conectou não and/or foi validado dentro do limite de tempo limite (60 segundos). Digite: <code>String</code> Obrigatório: não
NextToken	O token que indica o início da próxima página de resultados. Para especificar o início do conjunto de resultados, não forneça um valor. Se fornecer um ID de sessão do jogador, esse parâmetro será ignorado. Digite: <code>String</code> Obrigatório: não
Limite	O número máximo de resultados a serem retornados. Se fornecer um ID de sessão do jogador, esse parâmetro será ignorado. Digite: <code>int</code> Obrigatório: não

StopMatchBackfillRequest

Informações usadas para cancelar uma solicitação de alocação de marcação de jogos. O servidor do jogo comunica essas informações ao Amazon GameLift Servers serviço em uma [StopMatchBackfill\(\)](#) chamada.

Propriedades	Descrição
GameSessionArn	O identificador de sessão de jogo exclusivo da solicitação que está sendo cancelada. Digite: <code>string</code> Obrigatório: Sim
MatchmakingConfigurationArn	O identificador exclusivo do marcador de jogos para o qual essa solicitação foi enviada. Digite: <code>string</code> Obrigatório: Sim
TicketId	O identificador exclusivo do tíquete de solicitação de alocação a ser cancelado. Digite: <code>string</code> Obrigatório: Sim

GetFleetRoleCredentialsRequest

Esse tipo de dados dá ao servidor do jogo acesso limitado aos seus outros AWS recursos. Para obter mais informações, consulte, [Configure uma função de serviço do IAM para Amazon GameLift Servers](#).

Propriedades	Descrição
RoleArn	O Amazon Resource Name (ARN) da função de serviço que estende o acesso limitado aos seus AWS recursos.

Propriedades	Descrição
	Digite: <code>string</code>
	Obrigatório: Sim
RoleSessionName	O nome da sessão que descreve o uso das credenciais da função.
	Digite: <code>string</code>
	Obrigatório: não

AttributeValue

Use esses valores em pares de chave-valor de atributo [Jogador](#). Esse objeto permite especificar um valor de atributo usando qualquer um dos tipos de dados válidos: `string`, número, matriz de `string` ou mapa de dados. Cada objeto `AttributeValue` pode usar somente uma das propriedades disponíveis.

Propriedades	Descrição
attrType	Especifica o tipo de valor do atributo. Tipo: Um <code>AttrType</code> valor enum .. Obrigatório: não
S	Representa um valor de atributo de <code>string</code> . Digite: <code>string</code> Obrigatório: Sim
N	Representa um valor de atributo numérico. Digite: <code>double</code> Obrigatório: Sim
SL	Representa uma matriz de valores de atributos de <code>string</code> .

Propriedades	Descrição
	Digite: <code>string[]</code> Obrigatório: Sim
SDM	Representa um dicionário de chaves de string e valores duplos. Digite: <code>Dictionary<string, double></code> Obrigatório: Sim

AwsStringOutcome

Esse tipo de dados resulta de uma ação e produz um objeto com as seguintes propriedades:

Propriedades	Descrição
Resultado	O resultado da ação. Digite: <code>string</code> Obrigatório: não
Bem-sucedida	Se a ação foi bem-sucedida ou não. Digite: <code>bool</code> Obrigatório: Sim
Erro	O erro que ocorreu se a ação não foi bem-sucedida. Digite: the section called “GameLiftError” Obrigatório: não

GenericOutcome

Esse tipo de dados resulta de uma ação e produz um objeto com as seguintes propriedades:

Propriedades	Descrição
Bem-sucedida	Se a ação foi bem-sucedida ou não. Digite: bool Obrigatório: Sim
Erro	O erro que ocorreu se a ação não foi bem-sucedida. Digite: the section called “GameLiftError” Obrigatório: não

DescribePlayerSessionsOutcome

Esse tipo de dados resulta de uma ação e produz um objeto com as seguintes propriedades:

Propriedades	Descrição
Resultado	O resultado da ação. Digite: the section called “DescribePlayerSessionsResult” Obrigatório: não
Bem-sucedida	Se a ação foi bem-sucedida ou não. Digite: bool Obrigatório: Sim
Erro	O erro que ocorreu se a ação não foi bem-sucedida. Digite: the section called “GameLiftError” Obrigatório: não

DescribePlayerSessionsResult

Propriedades	Descrição
NextToken	O token indicando o início da próxima página de resultados. Para especificar o início do conjunto de resultados, não forneça um valor. Se fornecer um ID de sessão do jogador, esse parâmetro será ignorado. Digite: <code>string</code> Obrigatório: Sim
PlayerSessions	Uma coleção de objetos contendo propriedades para cada sessão de jogador que corresponde à solicitação. Digite: <code>IList<the section called "PlayerSession"></code> Obrigatório:
Bem-sucedida	Se a ação foi bem-sucedida ou não. Digite: <code>bool</code> Obrigatório: Sim
Erro	O erro que ocorreu se a ação não foi bem-sucedida. Digite: the section called "GameLiftError" Obrigatório: não

PlayerSession

Propriedades	Descrição
CreationTime	Digite: <code>long</code> Obrigatório: Sim

Propriedades	Descrição
FleetId	Digite: <code>string</code> Obrigatório: Sim
GameSessionId	Digite: <code>string</code> Obrigatório: Sim
IpAddress	Digite: <code>string</code> Obrigatório: Sim
PlayerData	Digite: <code>string</code> Obrigatório: Sim
PlayerId	Digite: <code>string</code> Obrigatório: Sim
PlayerSessionId	Digite: <code>string</code> Obrigatório: Sim
Porta	Digite: <code>int</code> Obrigatório: Sim
Status	Tipo: Um <code>PlayerSessionStatus</code> enum.. Obrigatório: Sim
TerminationTime	Digite: <code>long</code> Obrigatório: Sim
DnsName	Digite: <code>string</code> Obrigatório: Sim

StartMatchBackfillOutcome

Esse tipo de dados resulta de uma ação e produz um objeto com as seguintes propriedades:

Propriedades	Descrição
Resultado	O resultado da ação. Digite: the section called “StartMatchBackfillResult” Obrigatório: não
Bem-sucedida	Se a ação foi bem-sucedida ou não. Digite: bool Obrigatório: Sim
Erro	O erro que ocorreu se a ação não foi bem-sucedida. Digite: the section called “GameLiftError” Obrigatório: não

StartMatchBackfillResult

Propriedades	Descrição
TicketId	Digite: string Obrigatório: Sim

GetComputeCertificateOutcome

Esse tipo de dados resulta de uma ação e produz um objeto com as seguintes propriedades:

Propriedades	Descrição
Resultado	O resultado da ação.

Propriedades	Descrição
	Digite: the section called “GetComputeCertificateResult” Obrigatório: não
Bem-sucedida	Se a ação foi bem-sucedida ou não. Digite: bool Obrigatório: Sim
Erro	O erro que ocorreu se a ação não foi bem-sucedida. Digite: the section called “GameLiftError” Obrigatório: não

GetComputeCertificateResult

O caminho para o certificado TLS em seu computador e o nome do host do computador.

Propriedades	Descrição
CertificatePath	Digite: string Obrigatório: Sim
ComputeName	Digite: string Obrigatório: Sim

GetFleetRoleCredentialsOutcome

Esse tipo de dados resulta de uma ação e produz um objeto com as seguintes propriedades:

Propriedades	Descrição
Resultado	O resultado da ação.

Propriedades	Descrição
	Digite: the section called “GetFleetRoleCredentialsResult” Obrigatório: não
Bem-sucedida	Se a ação foi bem-sucedida ou não. Digite: <code>bool</code> Obrigatório: Sim
Erro	O erro que ocorreu se a ação não foi bem-sucedida. Digite: the section called “GameLiftError” Obrigatório: não

GetFleetRoleCredentialsResult

Propriedades	Descrição
AccessKeyId	O ID da chave de acesso para autenticar e fornecer acesso aos seus recursos AWS . Digite: <code>string</code> Obrigatório: não
AssumedRoleId	O ID do usuário ao qual o perfil de serviço pertence. Digite: <code>string</code> Obrigatório: não
AssumedRoleUserArn	O nome do recurso da Amazon (ARN) do usuário ao qual pertence o perfil de serviço. Digite: <code>string</code> Obrigatório: não

Propriedades	Descrição
Expiração	A quantidade de tempo até que suas credenciais de sessão expirem. Digite: <code>DateTime</code> Obrigatório: não
SecretAccessKey	O ID da chave de acesso secreta para autenticação. Digite: <code>string</code> Obrigatório: não
SessionToken	Um token para identificar a sessão ativa atual interagindo com seus AWS recursos. Digite: <code>string</code> Obrigatório: não
Bem-sucedida	Se a ação foi bem-sucedida ou não. Digite: <code>bool</code> Obrigatório: Sim
Erro	O erro que ocorreu se a ação não foi bem-sucedida. Digite: the section called “GameLiftError” Obrigatório: não

AwsDateTimeOutcome

Esse tipo de dados resulta de uma ação e produz um objeto com as seguintes propriedades:

Propriedades	Descrição
Resultado	O resultado da ação.

Propriedades	Descrição
	Digite: <code>DateTime</code> Obrigatório: não
Bem-sucedida	Se a ação foi bem-sucedida ou não. Digite: <code>bool</code> Obrigatório: Sim
Erro	O erro que ocorreu se a ação não foi bem-sucedida. Digite: the section called “GameLiftError” Obrigatório: não

GameLiftError

Propriedades	Descrição
ErrorType	O tipo de erro. Tipo: Um <code>GameLiftErrorType</code> enum .. Obrigatório: não
ErrorMessage	O nome do erro. Digite: <code>string</code> Obrigatório: não
ErrorMessage	A mensagem de erro. Digite: <code>string</code> Obrigatório: não

Enumerações

Os enums definidos para o SDK do servidor para Amazon GameLift Servers (C#) são definidos da seguinte forma:

AttrType

- NONE
- STRING
- DOUBLE
- STRING_LIST
- STRING_DOUBLE_MAP

GameLiftErrorType

Valor da string indicando o tipo de erro. Os valores válidos são:

- SERVICE_CALL_FAILED — Uma chamada para um serviço falhou. AWS
- LOCAL_CONNECTION_FAILED — A conexão local falhou. Amazon GameLift Servers
- NETWORK_NOT_INITIALIZED – A rede não foi inicializada.
- GAMESESSION_ID_NOT_SET – O ID da sessão do jogo não foi definido.
- BAD_REQUEST_EXCEPTION
- INTERNAL_SERVICE_EXCEPTION
- ALREADY_INITIALIZED — O Amazon GameLift Servers servidor ou cliente já foi inicializado com Initialize ().
- FLEET_MISMATCH – A frota de destino não corresponde à frota de uma GameSession ou PlayerSession.
- GAMELIFT_CLIENT_NOT_INITIALIZED — O cliente não foi inicializado. Amazon GameLift Servers
- GAMELIFT_SERVER_NOT_INITIALIZED — O servidor não foi inicializado. Amazon GameLift Servers
- GAME_SESSION_ENDED_FAILED — o SDK do servidor não conseguiu entrar em contato com o serviço para informar Amazon GameLift Servers que a sessão do jogo foi encerrada.
- GAME_SESSION_NOT_READY — A sessão de jogo do servidor não foi ativadaAmazon GameLift Servers.

- `GAME_SESSION_READY_FAILED` — o SDK do servidor não conseguiu entrar em contato com o serviço para informar Amazon GameLift Servers que a sessão do jogo está pronta.
- `INITIALIZATION_MISMATCH` – Um método de cliente foi chamado depois de `Server::Initialize()`, ou vice-versa.
- `NOT_INITIALIZED` — O Amazon GameLift Servers servidor ou cliente não foi inicializado com `Initialize ()`.
- `NO_TARGET_ALIASID_SET` – Um `aliasID` de destino não foi definido.
- `NO_TARGET_FLEET_SET` – Uma frota alvo não foi definida.
- `PROCESS_ENDING_FAILED` — o SDK do servidor não conseguiu entrar em contato com o serviço para Amazon GameLift Servers informar que o processo está sendo encerrado.
- `PROCESS_NOT_ACTIVE` — O processo do servidor ainda não está ativo, não está vinculado a um e não pode `GameSession` aceitar ou processar. `PlayerSessions`
- `PROCESS_NOT_READY` – O processo do servidor ainda não está pronto para ser ativado.
- `PROCESS_READY_FAILED` — o SDK do servidor não conseguiu entrar em contato com o serviço para informar que o processo está pronto.
- `SDK_VERSION_DETECTION_FAILED` – Falha na detecção da versão do SDK.
- `STX_CALL_FAILED` — Uma chamada para o componente de back-end do XStx servidor falhou.
- `STX_INITIALIZATION_FAILED` — O componente de back-end do XStx servidor falhou ao inicializar.
- `UNEXPECTED_PLAYER_SESSION` – Uma sessão de jogador não registrada foi encontrada pelo servidor.
- `WEBSOCKET_CONNECT_FAILURE`
- `WEBSOCKET_CONNECT_FAILURE_FORBIDDEN`
- `WEBSOCKET_CONNECT_FAILURE_INVALID_URL`
- `WEBSOCKET_CONNECT_FAILURE_TIMEOUT`
- `WEBSOCKET_RETRIABLE_SEND_MESSAGE_FAILURE` — Falha recuperável ao enviar uma mensagem ao Serviço. GameLift WebSocket
- `WEBSOCKET_SEND_MESSAGE_FAILURE` — Falha ao enviar uma mensagem para o serviço. GameLift WebSocket
- `MATCH_BACKFILL_REQUEST_VALIDATION` – A validação da solicitação falhou.
- `PLAYER_SESSION_REQUEST_VALIDATION` – A validação da solicitação falhou.

PlayerSessionCreationPolicy

Valor de string que indica se a sessão do jogo aceita novos jogadores. Os valores válidos são:

- ACCEPT_ALL – Aceite todas as novas sessões de jogador.
- DENY_ALL – Recuse todas as novas sessões de jogador.
- NOT_SET – A sessão do jogo não está configurada para aceitar ou negar sessões de novos jogadores.

PlayerSessionStatus

- ACTIVE
- COMPLETED
- NOT_SET
- RESERVED
- TIMEDOUT

[SDK 5.x do servidor C# para Amazon GameLift Servers -- Tipos de dados](#)

Tópicos

- [GetSdkVersion\(\)](#)
- [InitSDK\(\)](#)
- [InitSDK\(\)](#)
- [ProcessReady\(\)](#)
- [ProcessEnding\(\)](#)
- [ActivateGameSession\(\)](#)
- [UpdatePlayerSessionCreationPolicy\(\)](#)
- [GetGameSessionId\(\)](#)
- [GetTerminationTime\(\)](#)
- [AcceptPlayerSession\(\)](#)
- [RemovePlayerSession\(\)](#)
- [DescribePlayerSessions\(\)](#)
- [StartMatchBackfill\(\)](#)
- [StopMatchBackfill\(\)](#)

- [GetComputeCertificate\(\)](#)
- [GetFleetRoleCredentials\(\)](#)
- [Destroy\(\)](#)

GetSdkVersion()

Retorna o número da versão atual do SDK compilado no processo de servidor.

Sintaxe

```
AwsStringOutcome GetSdkVersion();
```

Valor de retorno

Se bem-sucedido, retornará a versão do SDK atual como um objeto [the section called “AwsStringOutcome”](#). A string retornada inclui o número da versão (exemplo, 5.0.0). Se não for bem-sucedido, retornará uma mensagem de erro.

Exemplo

```
var getSdkVersionOutcome = GameLiftServerAPI.GetSdkVersion();
```

InitSDK()

Inicializa o Amazon GameLift Servers SDK para uma frota gerenciada EC2 . Chame esse método na inicialização, antes que qualquer outra inicialização relacionada Amazon GameLift Servers ocorra. Esse método lê os parâmetros do servidor do ambiente host para configurar a comunicação entre o servidor e o Amazon GameLift Servers serviço.

Sintaxe

```
GenericOutcome InitSDK();
```

Valor de retorno

Se for bem-sucedido, retornará um InitSdkOutcome objeto para indicar que o processo do servidor está pronto para ser chamado [ProcessReady\(\)](#).

Exemplo

```
//Call InitSDK to establish a local connection with the GameLift agent to enable  
further communication.  
GenericOutcome initSDKOutcome = GameLiftServerAPI.InitSDK();
```

InitSDK()

Inicializa o Amazon GameLift Servers SDK para uma frota Anywhere. Chame esse método na inicialização, antes que qualquer outra inicialização relacionada Amazon GameLift Servers ocorra. Esse método requer parâmetros explícitos do servidor para configurar a comunicação entre o servidor e o Amazon GameLift Servers serviço.

Sintaxe

```
GenericOutcome InitSDK(ServerParameters serverParameters);
```

Parâmetros

ServerParameters

Para inicializar um servidor de jogo em uma frota Amazon GameLift Servers Anywhere, construa um `ServerParameters` objeto com as seguintes informações:

- O URL do WebSocket usado para se conectar ao seu servidor de jogo.
- O ID do processo usado para hospedar o servidor de jogos.
- O ID do computador que hospeda os processos do seu servidor de jogos.
- O ID da Amazon GameLift Servers frota que contém sua computação Amazon GameLift Servers Anywhere.
- O token de autorização gerado pela Amazon GameLift Servers operação.

Valor de retorno

Se for bem-sucedido, retornará um `InitSdkOutcome` objeto para indicar que o processo do servidor está pronto para ser chamado [ProcessReady\(\)](#).

Note

Se as chamadas para `InitSDK()` estiverem falhando para compilações de jogos implantadas em frotas do Anywhere, verifique o parâmetro `ServerSdkVersion` usado ao

criar o recurso de compilação. Você deve definir explicitamente esse valor para a versão do SDK do servidor em uso. O valor padrão desse parâmetro é 4.x, o que não é compatível. Para resolver esse problema, crie uma nova versão e implante-a em uma nova frota.

Exemplo

```
//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 que o processo do servidor está pronto para hospedar sessões de jogo. Chame esse método após invocar [InitSDK\(\)](#). Esse método deve ser chamado somente uma vez por processo.

Sintaxe

```
GenericOutcome ProcessReady(ProcessParameters processParameters)
```

Parâmetros

[ProcessParameters](#)

Um objeto `ProcessParameters` contém informações sobre o processo do servidor.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

Este exemplo ilustra as implementações do método e da função delegada.

```
// 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 que o processo do servidor está sendo encerrado. Chame esse método depois de todas as outras tarefas de limpeza (incluindo o encerramento da sessão ativa do jogo) e antes de encerrar o processo. Dependendo do resultado `ProcessEnding()`, o processo sai com sucesso (0) ou erro (-1) e gera um evento de frota. Se o processo for encerrado com um erro, o evento de frota gerado será `SERVER_PROCESS_TERMINATED_UNHEALTHY`.

Sintaxe

```
GenericOutcome ProcessEnding()
```

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

Este exemplo chama `ProcessEnding()` e `Destroy()` antes de encerrar o processo do servidor com um código de saída bem-sucedido ou de erro.

```
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 que o processo do servidor ativou uma sessão de jogo e agora está pronto para receber conexões de jogadores. Essa ação deve ser chamada como parte da função de retorno de chamada `onStartGameSession()`, após toda inicialização da sessão do jogo.

Sintaxe

```
GenericOutcome ActivateGameSession()
```

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

Este exemplo mostra `ActivateGameSession()` sendo chamado como parte da função de delegação `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()

Atualiza a capacidade da sessão do jogo atual para aceitar novas sessões de jogador. Atualiza a capacidade da sessão do jogo atual para aceitar novas sessões de jogador.

Sintaxe

```
GenericOutcome UpdatePlayerSessionCreationPolicy(PlayerSessionCreationPolicy  
playerSessionPolicy)
```

Parâmetros

playerSessionPolicy

Valor de string que indica se a sessão do jogo aceita novos jogadores.

Os valores válidos são:

- ACCEPT_ALL – Aceite todas as novas sessões de jogador.
- DENY_ALL – Recuse todas as novas sessões de jogador.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

Este exemplo define a política de ingresso da sessão do jogo atual para aceitar todos os jogadores.

```
GenericOutcome updatePlayerSessionPolicyOutcome =  
GameLiftServerAPI.UpdatePlayerSessionCreationPolicy(PlayerSessionCreationPolicy.ACCEPT_ALL);
```

GetGameSessionId()

Recupera o ID da sessão de jogo hospedada pelo processo do servidor ativo.

Para processos inativos que não são ativados com uma sessão de jogo, a chamada retorna um [the section called “GameLiftError”](#).

Sintaxe

```
AwsStringOutcome GetGameSessionId()
```

Valor de retorno

Se bem-sucedido, retornará o ID da sessão de jogo como um objeto [the section called “AwsStringOutcome”](#). Se não for bem-sucedido, retornará uma mensagem de erro."

Exemplo

```
AwsStringOutcome getGameSessionIdOutcome = GameLiftServerAPI.GetGameSessionId();
```

GetTerminationTime()

Retorna a hora em que um processo do servidor está programado para ser desligado, se essa informação estiver disponível. Um processo do servidor executa essa ação depois de receber um `onProcessTerminate()` retorno de chamada de Amazon GameLift Servers. Amazon GameLift Servers chamadas `onProcessTerminate()` pelos seguintes motivos:

- Quando o processo do servidor relatou problemas de saúde ou não respondeu Amazon GameLift Servers.
- Ao encerrar a instância durante um evento de redução.
- Quando uma instância é encerrada devido a uma [interrupção na instância spot](#).

Sintaxe

```
AwsDateTimeOutcome GetTerminationTime()
```

Valor de retorno

Se for bem-sucedido, retornará o horário de término como um objeto [the section called “AwsDateTimeOutcome”](#). O valor é o tempo de rescisão, expresso em tiques decorridos desde então 0001 00:00:00. Por exemplo, o valor da data e hora 2020-09-13 12:26:40 -000Z é igual aos tiques 637355968000000000. Se nenhum horário de rescisão estiver disponível, o retornará uma mensagem de erro.

Exemplo

```
AwsDateTimeOutcome getTerminationTimeOutcome = GameLiftServerAPI.GetTerminationTime();
```

AcceptPlayerSession()

Notifica Amazon GameLift Servers que um jogador com o ID de sessão de jogador especificado se conectou ao processo do servidor e precisa de validação. Amazon GameLift Servers verifica se o ID da sessão do jogador é válido. Depois que a sessão do jogador for validada, o status da vaga do jogador é Amazon GameLift Servers alterado de RESERVADO para ATIVO.

Sintaxe

```
GenericOutcome AcceptPlayerSession(String playerId)
```

Parâmetros

playerSessionId

ID exclusivo emitido GameLift quando uma nova sessão de jogador é criada.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

Este exemplo ilustra uma função para lidar com uma solicitação de conexão, incluindo a validação e a rejeição de uma sessão de jogador inválida. 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 que um jogador se desconectou do processo do servidor. Em resposta, Amazon GameLift Servers altera o slot do jogador para disponível.

Sintaxe

```
GenericOutcome RemovePlayerSession(String playerId)
```

Parâmetros

playerSessionId

ID exclusivo emitido Amazon GameLift Servers quando uma nova sessão de jogador é criada.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

```
GenericOutcome removePlayerSessionOutcome =  
    GameLiftServerAPI.RemovePlayerSession(playerSessionId);
```

DescribePlayerSessions()

Recupera dados da sessão do jogador, que incluem configurações, metadados da sessão e dados do jogador. Use essa ação para obter informações de uma única sessão de jogador, para todas as sessões de jogador em uma sessão de jogo, ou para todas as sessões de jogador associadas a um único ID de jogador.

Sintaxe

```
DescribePlayerSessionsOutcome DescribePlayerSessions(DescribePlayerSessionsRequest  
    describePlayerSessionsRequest)
```

Parâmetros

[DescribePlayerSessionsRequest](#)

Um objeto [the section called “DescribePlayerSessionsRequest”](#) que descreve quais sessões de jogador recuperar.

Valor de retorno

Se bem-sucedido, retorna um objeto [the section called “DescribePlayerSessionsOutcome”](#) que contém um conjunto de objetos de sessão do jogador que atendem aos parâmetros de solicitação.

Exemplo

Este exemplo ilustra uma solicitação para todas as sessões de jogador conectadas ativamente a uma sessão de jogo especificada. Ao omitir NextToken e definir o valor limite como 10, Amazon GameLift Servers retornará os primeiros registros da sessão de 10 jogadores que correspondam à solicitação.

```
// 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()

Envia uma solicitação para encontrar novos jogadores para os slots abertos em uma sessão de jogo criada com o FlexMatch. Para obter mais informações, consulte o [recurso FlexMatch de preenchimento](#).

Esta ação é assíncrona. Se novos jogadores forem combinados, Amazon GameLift Servers fornece dados atualizados do matchmaker usando a função de retorno de chamada.

`OnUpdateGameSession()`

Um processo de servidor pode ter apenas uma solicitação de alocação de correspondência ativa por vez. Para enviar uma nova solicitação, primeiro chame [StopMatchBackfill\(\)](#) para cancelar a solicitação original.

Sintaxe

```
StartMatchBackfillOutcome StartMatchBackfill (StartMatchBackfillRequest  
startBackfillRequest);
```

Parâmetros

[StartMatchBackfillRequest](#)

Um objeto `StartMatchBackfillRequest` contém informações sobre a solicitação de alocação.

Valor de retorno

Retorna um objeto [the section called “StartMatchBackfillOutcome”](#) com o ID do tíquete de alocação de correspondência ou um falha com uma mensagem de erro.

Exemplo

```
// 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()

Cancela uma solicitação de alocação de correspondência ativa. Para obter mais informações, consulte o [recurso FlexMatch de preenchimento](#).

Sintaxe

```
GenericOutcome StopMatchBackfill (StopMatchBackfillRequest stopBackfillRequest);
```

Parâmetros

[StopMatchBackfillRequest](#)

Um objeto `StopMatchBackfillRequest` que fornece detalhes sobre o tíquete de marcação de jogos que você está interrompendo.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

```
// 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 o caminho para o certificado TLS usado para criptografar a conexão de rede entre o servidor do jogo e o cliente do jogo. Você pode usar o caminho do certificado ao registrar seu dispositivo computacional Amazon GameLift Servers em uma frota Anywhere. Para obter mais informações, consulte, [RegisterCompute](#).

Sintaxe

```
GetComputeCertificateOutcome GetComputeCertificate();
```

Valor de retorno

Retorna um `GetComputeCertificateResponse` objeto que contém o seguinte:

- **CertificatePath:** o caminho para o certificado TLS em seu recurso computacional. Ao usar uma frota Amazon GameLift Servers gerenciada, esse caminho contém:
 - `certificate.pem`: o certificado do usuário final. A cadeia completa de certificados é a combinação de `certificateChain.pem` anexados a esse certificado.
 - `certificateChain.pem`: a cadeia de certificados que contém o certificado raiz e os certificados intermediários.
 - `rootCertificate.pem`: o certificado raiz.
 - `privateKey.pem`: a chave privada para o certificado do usuário final.
- **ComputeName:** o nome do seu recurso computacional.

Exemplo

```
GetComputeCertificateOutcome getComputeCertificateOutcome =
    GameLiftServerAPI.GetComputeCertificate();
```

GetFleetRoleCredentials()

Recupera as credenciais da função do IAM que autorizam Amazon GameLift Servers a interação com outras pessoas. Serviços da AWS Para obter mais informações, consulte [Comunique-se com outros AWS recursos de suas frotas](#).

Sintaxe

```
GetFleetRoleCredentialsOutcome GetFleetRoleCredentials(GetFleetRoleCredentialsRequest request);
```

Parâmetros

[GetFleetRoleCredentialsRequest](#)

Credenciais de função que ampliam o acesso limitado aos seus AWS recursos no servidor do jogo.

Valor de retorno

Informa um objeto [the section called “GetFleetRoleCredentialsOutcome”](#).

Exemplo

```
// form the fleet credentials request
GetFleetRoleCredentialsRequest getFleetRoleCredentialsRequest = new
    GetFleetRoleCredentialsRequest(){
    RoleArn = "arn:aws:iam::123456789012:role/service-role/exampleGameLiftAction"
};
GetFleetRoleCredentialsOutcome GetFleetRoleCredentialsOutcome credentials =
    GetFleetRoleCredentials(getFleetRoleCredentialsRequest);
```

Destroy()

Libera o SDK Amazon GameLift Servers do servidor de jogos da memória. Como melhor prática, chame esse método antes `ProcessEnding()` e depois de encerrar o processo. Se você estiver usando uma frota Anywhere e não estiver encerrando os processos do servidor após cada sessão de jogo, ligue `Destroy()` e reinicialize antes de notificar Amazon GameLift Servers que o processo está pronto para hospedar uma sessão de jogo. `InitSDK()` `ProcessReady()`

Sintaxe

```
GenericOutcome Destroy()
```

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

```
// 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);
}
```

SDK do servidor Go para Amazon GameLift Servers -- Ações

Use a referência 5.x do SDK do servidor para integrar seu jogo multijogador para hospedagem com Amazon GameLift Servers. Para obter orientação sobre o processo de integração, consulte [Adicione Amazon GameLift Servers ao seu servidor de jogos](#).

`GameLiftServerAPI.go` define as ações do SDK do servidor Go.

SDK do servidor Go para Amazon GameLift Servers -- Tipos de dados

Use a referência do SDK do servidor para integrar seu jogo multijogador para hospedagem com Amazon GameLift Servers. Para obter orientação sobre o processo de integração, consulte [Adicione Amazon GameLift Servers ao seu servidor de jogos](#).

GameLiftServerAPI.go define as ações do SDK do servidor Go.

[SDK do servidor Go para Amazon GameLift Servers -- Ações](#)

Tipos de dados

- [LogParameters](#)
- [ProcessParameters](#)
- [UpdateGameSession](#)
- [GameSession](#)
- [ServerParameters](#)
- [StartMatchBackfillRequest](#)
- [Jogador](#)
- [DescribePlayerSessionsRequest](#)
- [StopMatchBackfillRequest](#)
- [GetFleetRoleCredentialsRequest](#)

LogParameters

Um objeto que identifica arquivos gerados durante uma sessão de jogo que você Amazon GameLift Servers deseja carregar e armazenar após o término da sessão de jogo. O servidor do jogo fornece LogParameters o Amazon GameLift Servers como parte de um ProcessParameters objeto em uma [ProcessReady\(\)](#) chamada.

Properties	Descrição
LogPaths	A lista de caminhos de diretório para os arquivos de log do servidor de jogos que você Amazon GameLift Servers deseja armazenar para acesso futuro. O processo do servidor gera esses arquivos durante cada sessão de jogo. Você define caminhos e nomes de arquivos em seu servidor de jogo e os armazena no diretório raiz de construção do jogo. Os caminhos de log devem ser absolutos. Por exemplo, se a compilação do jogo armazenasse logs de sessão de jogo em um caminho como MyGame\sessionLogs\ , o caminho estará em c:\game\MyGame\sessionLogs uma instância do Windows.

	Digite: <code>[]string</code>
	Obrigatório: não

ProcessParameters

Um objeto que descreve a comunicação entre um processo do servidor Amazon GameLift Servers e o processo do servidor. O processo do servidor fornece essas informações Amazon GameLift Servers com uma chamada para [ProcessReady\(\)](#).

Properties	Descrição
LogParameters	Um objeto com caminhos de diretório para arquivos que são gerados durante uma sessão de jogo. Amazon GameLift Servers copia e armazena os arquivos para acesso futuro. Digite: LogParameters Obrigatório: não
OnHealthCheck	A função de retorno de chamada que Amazon GameLift Servers invoca para solicitar um relatório de status de saúde do processo do servidor. Amazon GameLift Servers chama essa função a cada 60 segundos e espera 60 segundos por uma resposta. O processo do servidor retorna TRUE se estiver íntegro, FALSE se não estiver íntegro. Se nenhuma resposta for retornada, Amazon GameLift Servers registra o processo do servidor como não íntegro. Digite: <code>OnHealthCheck func() bool</code> Obrigatório: não
OnProcessTerminate	A função de retorno de chamada que Amazon GameLift Servers invoca para forçar o encerramento do processo do servidor. Depois de chamar essa função, Amazon GameLift Servers aguarda 5 minutos até que o processo do servidor seja encerrado e responda com uma ProcessEnding() chamada antes de encerrar o processo do servidor. Digite: <code>OnProcessTerminate func()</code>

	Obrigatório: Sim
<code>OnStartGameSession</code>	A função de retorno de chamada que Amazon GameLift Servers invoca para passar um objeto de sessão de jogo atualizado para o processo do servidor. Amazon GameLift Servers chama essa função quando uma solicitação de preenchimento de correspondência é processada para fornecer dados atualizados do matchmaker. Ele transmite um objeto GameSession, uma atualização de status (<code>updateReason</code>) e o ID do tíquete de alocação de correspondência. Digite: <code>OnStartGameSession func (model.GameSession)</code> Obrigatório: Sim
<code>OnUpdateGameSession</code>	A função de retorno de chamada que Amazon GameLift Servers invoca para passar informações atualizadas da sessão do jogo para o processo do servidor. Amazon GameLift Servers chama essa função após processar uma solicitação de preenchimento de correspondência para fornecer dados atualizados do matchmaker. Digite: <code>OnUpdateGameSession func (model.UpdateGameSession)</code> Obrigatório: não
<code>Port</code>	O número da porta em que o processo de servidor escuta para novas conexões de jogador. O valor deve estar dentro do intervalo de portas configurado para qualquer frota que implanta essa compilação do servidor de jogo. Esse número de porta está incluído nos objetos das sessões de jogo e jogador, que as sessões de jogo usam ao se conectar a um processo de servidor. Digite: <code>int</code> Obrigatório: Sim

UpdateGameSession

As atualizações em um objeto de sessão de jogo, que incluem o motivo pelo qual a sessão de jogo foi atualizada e o ID do tíquete de alocação relacionado, caso o preenchimento esteja sendo usado para preencher sessões de jogadores na sessão de jogo.

Propriedades	Descrição
GameSession	Um objeto GameSession. O objeto <code>GameSession</code> contém propriedades que descrevem uma sessão de jogo. Digite: <code>GameSession GameSession()</code> Obrigatório: Sim
UpdateReason	O motivo pelo qual a sessão do jogo está sendo atualizada. Digite: <code>UpdateReason UpdateReason()</code> Obrigatório: Sim
BackfillTicketId	O ID do tíquete de alocação que está tentando atualizar a sessão do jogo. Digite: <code>String</code> Obrigatório: não

GameSession

Os detalhes de uma sessão de jogo.

Propriedades	Descrição
GameSessionId	Um identificador exclusivo da sessão de jogo. Um nome do recurso da Amazon (ARN) de sessão de jogo tem o seguinte formato: <code>arn:aws:gamelift:<region>::gamesession/<fleet ID>/<custom ID string or idempotency token></code> . Digite: <code>String</code>

Propriedades	Descrição
	Obrigatório: não
Name	Um rótulo descritivo da sessão do jogo. Digite: <code>String</code> Obrigatório: não
FleetId	Um identificador exclusivo da frota em que a sessão de jogo está sendo executada. Digite: <code>String</code> Obrigatório: não
MaximumPlayerSessionCount	O número máximo de conexões do jogador com a sessão do jogo. Digite: <code>Integer</code> Obrigatório: não
Porta	O número da porta da sessão de jogo. Para se conectar a um servidor de Amazon GameLift Servers jogos, um aplicativo precisa do endereço IP e do número da porta. Digite: <code>Integer</code> Obrigatório: não
IpAddress	O endereço IP da sessão de jogo. Para se conectar a um servidor de Amazon GameLift Servers jogos, um aplicativo precisa do endereço IP e do número da porta. Digite: <code>String</code> Obrigatório: não

Propriedades	Descrição
GameSessionData	Um conjunto de propriedades de sessão de jogo personalizadas, formatadas como um único valor de string. Digite: <code>String</code> Obrigatório: não
MatchmakerData	As informações sobre o processo de matchmaking usado para criar a sessão do jogo, na sintaxe JSON, formatado como uma string. Além da configuração de matchmaking usada, ele contém dados sobre todos os jogadores atribuídos à partida, incluindo atributos do jogador e atribuições da equipe. Digite: <code>String</code> Obrigatório: não
GameProperties	Um conjunto de propriedades personalizadas para uma sessão de jogo, formatado como pares de chave:valor. Essas propriedades são passadas com uma solicitação de início de uma nova sessão de jogo. Digite: <code>map[string] string</code> Obrigatório: não

Propriedades	Descrição
DnsName	O identificador DNS atribuído à instância que está executando a sessão do jogo. Os valores têm o formato a seguir: - Frotas habilitadas para TLS: <code><unique identifier>.<region identifier>.amazongamelift.com</code> - Non-TLS-enabled frotas: <code>ec2-<unique identifier>.compute.amazonaws.com</code> Ao se conectar a uma sessão de jogo que está sendo executada em uma frota compatível com TLS, você deve usar o nome DNS, não o endereço IP. Digite: String Obrigatório: não

ServerParameters

Informações usadas para manter a conexão entre um servidor Amazon GameLift Servers Anywhere e o Amazon GameLift Servers serviço. Essas informações são usadas ao iniciar novos processos de servidor com [InitSDK\(\)](#). Para servidores hospedados em EC2 instâncias Amazon GameLift Servers gerenciadas, use um objeto vazio.

Propriedades	Descrição
WebSocket URL	Os <code>GameLiftServerSdkEndpoint</code> Amazon GameLift Servers retornos quando você RegisterCompute compra um recurso de computação Amazon GameLift Servers Anywhere. Digite: string Obrigatório: Sim
ProcessID	Um identificador exclusivo registrado no processo do servidor que hospeda seu jogo. Digite: string

Propriedades	Descrição
	Obrigatório: Sim
HostID	O identificador exclusivo do recurso de computação que está hospedando o novo processo de servidor. O HostID é o ComputeName usado quando você registrou seu computador. Para obter mais informações, consulte RegisterCompute. Digite: string Obrigatório: Sim
FleetID	O identificador exclusivo da frota na qual o computador está registrado. Para obter mais informações, consulte RegisterCompute. Digite: string Obrigatório: Sim
AuthToken	O token de autenticação gerado por Amazon GameLift Servers isso autentica seu servidor em Amazon GameLift Servers. Para obter mais informações, consulte GetComputeAuthToken. Digite: string Obrigatório: Sim

StartMatchBackfillRequest

Informações usadas para criar uma solicitação de alocação de marcação de jogos. O servidor do jogo comunica essas informações Amazon GameLift Servers em uma [StartMatchBackfill\(\)](#) chamada.

Propriedades	Descrição
GameSessionArn	O identificador da sessão de jogo exclusivo. A operação de API retorna GetGameSessionId o identificador no formato de ARN. Digite: String

Propriedades	Descrição
	Obrigatório: Sim
MatchmakingConfigurationArn	O identificador exclusivo (na forma de um ARN), para o marcador de jogos a ser usado para essa solicitação. O marcador de jogos da ARN da sessão de jogo original está no objeto de sessão de jogo, na propriedade de dados do marcador de jogos. Para obter mais informações sobre dados de matchmaking, consulte Trabalhar com dados de marcador de jogos. Digite: <code>String</code> Obrigatório: Sim
Jogadores	Um conjunto de dados que representa todos os jogadores que estão na sessão do jogo no momento. O marcador de jogos usa essas informações para pesquisar novos jogadores que sejam correspondências ideais para os jogadores atuais. Digite: <code>[]model.Player</code> Obrigatório: Sim
TicketId	O identificador exclusivo de um tíquete de solicitação de marcação de jogo ou de alocação de correspondência. Se você não fornecer um valor, Amazon GameLift Servers gera um. Use esse identificador para acompanhar o status do tíquete de alocação de correspondência ou cancelar a solicitação, se necessário. Digite: <code>String</code> Obrigatório: não

Jogador

O objeto que representa um jogador em marcação de jogos. Quando uma solicitação de matchmaking começa, um jogador tem um ID de jogador, atributos, e possivelmente dados de latência. Amazon GameLift Servers adiciona informações da equipe após a realização de uma partida.

Propriedades	Descrição
LatencyInMS	Um conjunto de valores expressos em milissegundos que indicam a quantidade de latência que um jogador experimenta quando conectado a um local. Se essa propriedade for usada, o player só será correspondido aos locais listados. Se um marcador de jogos tiver uma regra que avalie a latência do jogador, os jogadores deverão reportar a latência a ser correspondida. Digite: <code>map[string] int</code> Obrigatório: não
PlayerAttributes	Uma coleção de pares de chave:valor que contêm informações de jogadores para uso em marcação de jogos. As chaves de atributos do jogador devem corresponder às PlayerAttributes usadas em um conjunto de regras de matchmaking. Para obter mais informações sobre os atributos do jogador, consulte Attribute Value. Digite: <code>map[string] AttributeValue</code> Obrigatório: não
PlayerId	Um identificador exclusivo para um jogador. Digite: <code>String</code> Obrigatório: não
Equipe	O nome da equipe à qual o jogador está designado em uma partida. Você define o nome da equipe no conjunto de regras de marcação de jogos. Digite: <code>String</code> Obrigatório: não

DescribePlayerSessionsRequest

Um objeto que especifica quais sessões de jogador devem ser recuperadas. O processo do servidor fornece essas informações com uma [DescribePlayerSessions\(\)](#) chamada para Amazon GameLift Servers.

Propriedades	Descrição
GameSessionID	Um identificador da sessão de jogo exclusivo. Use esse parâmetro para solicitar todas as sessões de jogador da sessão de jogo especificada. O formato do ID da sessão do jogo é <code>arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string></code>. O <code>GameSessionID</code> é uma string de ID personalizada ou uma string gerada. Digite: String Obrigatório: não
PlayerSessionID	Um identificador exclusivo para uma sessão de jogador. Use esse parâmetro para solicitar uma única sessão de jogador específica. Digite: String Obrigatório: não
PlayerID	Um identificador exclusivo para um jogador. Use este parâmetro para solicitar todas as sessões de jogador para um jogador específico. Consulte Gerar jogador IDs. Digite: String Obrigatório: não
PlayerSessionStatusFilter	O status da sessão de jogador para filtrar resultados. Entre os status da sessão de jogador possíveis incluem: RESERVED - A solicitação de sessão do jogador foi recebida, mas o jogador não se conectou ao processo do servidor nem foi validado.

Propriedades	Descrição
	• ACTIVE - O jogador foi validado pelo processo do servidor e está conectado. • COMPLETED – A conexão do jogador foi interrompida. • TIMEDOUT – Uma solicitação de sessão de jogador foi recebida, mas o jogador não se conectou ou não foi validado dentro do tempo limite (60 segundos). Digite: <code>String</code> Obrigatório: não
<code>NextToken</code>	O token que indica o início da próxima página de resultados. Para especificar o início do conjunto de resultados, não forneça um valor. Se fornecer um ID de sessão do jogador, esse parâmetro será ignorado. Digite: <code>String</code> Obrigatório: não
<code>Limit</code>	O número máximo de resultados a serem retornados. Se fornecer um ID de sessão do jogador, esse parâmetro será ignorado. Digite: <code>int</code> Obrigatório: não

StopMatchBackfillRequest

Informações usadas para cancelar uma solicitação de alocação de marcação de jogos. O servidor do jogo comunica essas informações ao Amazon GameLift Servers serviço em uma [StopMatchBackfill\(\)](#) chamada.

Propriedades	Descrição
<code>GameSessionArn</code>	O identificador de sessão de jogo exclusivo da solicitação que está sendo cancelada.

Propriedades	Descrição
	Digite: <code>string</code> Obrigatório: não
MatchmakingConfigurationArn	O identificador exclusivo do marcador de jogos para o qual essa solicitação foi enviada. Digite: <code>string</code> Obrigatório: não
TicketId	O identificador exclusivo do tíquete de solicitação de alocação a ser cancelado. Digite: <code>string</code> Obrigatório: não

GetFleetRoleCredentialsRequest

As credenciais da função que estendem o acesso limitado aos seus AWS recursos no servidor do jogo. Para obter mais informações, consulte, [Configure uma função de serviço do IAM para Amazon GameLift Servers](#).

Propriedades	Descrição
RoleArn	O ARN do perfil de serviço que estende o acesso limitado aos seus recursos AWS . Digite: <code>string</code> Obrigatório: Sim
RoleSessionName	O nome da sessão que descreve o uso das credenciais da função. Digite: <code>string</code> Obrigatório: Sim

SDK do servidor Go para Amazon GameLift Servers -- Tipos de dados

Ações

- [GetSdkVersion\(\)](#)
- [InitSDK\(\)](#)
- [ProcessReady\(\)](#)
- [ProcessEnding\(\)](#)
- [ActivateGameSession\(\)](#)
- [UpdatePlayerSessionCreationPolicy\(\)](#)
- [GetGameSessionId\(\)](#)
- [GetTerminationTime\(\)](#)
- [AcceptPlayerSession\(\)](#)
- [RemovePlayerSession\(\)](#)
- [DescribePlayerSessions\(\)](#)
- [StartMatchBackfill\(\)](#)
- [StopMatchBackfill\(\)](#)
- [GetComputeCertificate\(\)](#)
- [GetFleetRoleCredentials\(\)](#)
- [Destroy\(\)](#)

GetSdkVersion()

Retorna o número da versão atual do SDK compilado no processo de servidor.

Sintaxe

```
func GetSdkVersion() (string, error)
```

Valor de retorno

Se bem-sucedido, retornará a versão do SDK como uma sequência. A string retornada inclui o número da versão (exemplo, 5.0.0). Se não for bem-sucedido, retornará uma mensagem de erro como `common.SdkVersionDetectionFailed`.

Exemplo

```
version, err := server.GetSdkVersion()
```

InitSDK()

Inicializa o Amazon GameLift Servers SDK. Chame esse método na inicialização antes que qualquer outra inicialização relacionada Amazon GameLift Servers ocorra. Esse método configura a comunicação entre o servidor e o Amazon GameLift Servers serviço.

Sintaxe

```
func InitSDK(params ServerParameters) error
```

Parâmetros

ServerParameters

Para inicializar um servidor de jogo em uma frota Amazon GameLift Servers Anywhere, construa um `ServerParameters` objeto com as seguintes informações:

- O URL do WebSocket usado para se conectar ao seu servidor de jogo.
- O ID do processo usado para hospedar o servidor de jogos.
- O ID do computador que hospeda os processos do seu servidor de jogos.
- O ID da Amazon GameLift Servers frota que contém sua computação Amazon GameLift Servers Anywhere.
- O token de autorização gerado pela Amazon GameLift Servers operação.

Para inicializar um servidor de jogo em uma EC2 frota Amazon GameLift Servers gerenciada, construa um `ServerParameters` objeto sem parâmetros. Com essa chamada, o Amazon GameLift Servers agente configura o ambiente computacional e se conecta automaticamente ao Amazon GameLift Servers serviço para você.

Valor de retorno

Se for bem-sucedido, retornará um erro `nil` para indicar que o processo do servidor está pronto para ser chamado [ProcessReady\(\)](#).

 Note

Se as chamadas para `InitSDK()` estiverem falhando para compilações de jogos implantadas em frotas do Anywhere, verifique o parâmetro `ServerSdkVersion` usado ao criar o recurso de compilação. Você deve definir explicitamente esse valor para a versão do SDK do servidor em uso. O valor padrão desse parâmetro é 4.x, o que não é compatível. Para resolver esse problema, crie uma nova versão e implante-a em uma nova frota.

Exemplo

Amazon GameLift ServersExemplo em qualquer lugar

```
//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 Servers EC2 exemplo gerenciado

```
//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 que o processo do servidor está pronto para hospedar sessões de jogo. Chame esse método após invocar [InitSDK\(\)](#). Esse método deve ser chamado apenas uma vez por processo.

Sintaxe

```
func ProcessReady(param ProcessParameters) error
```

Parâmetros

ProcessParameters

Um objeto [ProcessParameters](#) transmite as seguintes informações sobre o processo do servidor:

- Os nomes dos métodos de retorno de chamada implementados no código do servidor do jogo que o Amazon GameLift Servers serviço invoca para se comunicar com o processo do servidor.
- O número da porta em que o processo de servidor está escutando.
- O tipo de [LogParameters](#) dados que contém o caminho para qualquer arquivo específico da sessão de jogo que você deseja capturar e Amazon GameLift Servers armazenar.

Valor de retorno

Retorna um erro com uma mensagem de erro se o método falhar. Se o método for bem-sucedido, ele retornará `nil`.

Exemplo

Este exemplo ilustra as implementações das funções de chamada e delegação [ProcessReady\(\)](#).

```
// 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 que o processo do servidor está sendo encerrado. Chame esse método depois de todas as outras tarefas de limpeza (incluindo o encerramento da sessão ativa do jogo) e antes de encerrar o processo. Dependendo do resultado `ProcessEnding()`, o processo sai com sucesso (0) ou erro (-1) e gera um evento de frota. Se o processo for encerrado com um erro, o evento de frota gerado será `SERVER_PROCESS_TERMINATED_UNHEALTHY`.

Sintaxe

```
func ProcessEnding() error
```

Valor de retorno

Retorna um código de erro 0 ou um código de erro definido.

Exemplo

```
// 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 que o processo do servidor ativou uma sessão de jogo e agora está pronto para receber conexões de jogadores. Essa ação é chamada como parte da função de retorno de chamada `onStartGameSession()`, após toda a inicialização da sessão do jogo.

Sintaxe

```
func ActivateGameSession() error
```

Valor de retorno

Retorna um erro com uma mensagem de erro se o método falhar.

Exemplo

Este exemplo mostra chamado `ActivateGameSession()` como parte da função de delegação `onStartGameSession()`.

```
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()

Atualiza a capacidade da sessão do jogo atual para aceitar novas sessões de jogador. Atualiza a capacidade da sessão do jogo atual para aceitar novas sessões de jogador.

Sintaxe

```
func UpdatePlayerSessionCreationPolicy(policy model.PlayerSessionCreationPolicy) error
```

Parâmetros

playerSessionCreationPolítica

Valor de string que indica se a sessão do jogo aceita novos jogadores.

Os valores válidos são:

- **model.AcceptAll** – Aceite todas as novas sessões de jogador.
- **model.DenyAll** – Recuse todas as novas sessões de jogador.

Valor de retorno

Retorna um erro com uma mensagem de erro se ocorrer uma falha.

Exemplo

Este exemplo define a política de ingresso da sessão do jogo atual para aceitar todos os jogadores.


```
err := server.UpdatePlayerSessionCreationPolicy(model.AcceptAll)
```

GetGameSessionId()

Recupera o ID da sessão de jogo hospedada pelo processo do servidor ativo.

Sintaxe

```
func GetGameSessionID() (string, error)
```

Parâmetros

Essa ação não tem um parâmetro.

Valor de retorno

Se bem-sucedido, retornará o ID da sessão de jogo e nenhum erro. Para processos inativos que ainda não foram ativados com uma sessão de jogo, a chamada retorna uma string vazia e um erro `nil`.

Exemplo

```
gameSessionID, err := server.GetGameSessionID()
```

GetTerminationTime()

Retorna a hora em que um processo do servidor está programado para ser desligado se essa informação estiver disponível. Um processo do servidor executa essa ação após receber um `onProcessTerminate()` retorno de chamada de Amazon GameLift Servers. Amazon GameLift Servers chamadas `onProcessTerminate()` pelos seguintes motivos:

- Quando o processo do servidor relatou problemas de saúde ou não respondeu Amazon GameLift Servers.
- Ao encerrar a instância durante um evento de redução.
- Quando uma instância é encerrada devido a uma [interrupção na instância spot](#).

Sintaxe

```
func GetTerminationTime() (int64, error)
```

Valor de retorno

Se for bem-sucedido, retornará o registro de data e hora em segundos em que o processo do servidor está programado para ser encerrado e um erro `nil` de encerramento. O valor é o tempo de rescisão, expresso em tiques decorridos de 0001 00:00:00. Por exemplo, o valor da data e hora 2020-09-13 12:26:40 -000Z é igual aos tiques 637355968000000000. Se nenhum horário de rescisão estiver disponível, o retornará uma mensagem de erro.

Exemplo

```
terminationTime, err := server.GetTerminationTime()
```

AcceptPlayerSession()

Notifica Amazon GameLift Servers que um jogador com o ID de sessão de jogador especificado se conectou ao processo do servidor e precisa de validação. Amazon GameLift Servers verifica se o ID da sessão do jogador é válido. Depois que a sessão do jogador for validada, o status do slot do jogador será Amazon GameLift Servers alterado de RESERVED para ACTIVE.

Sintaxe

```
func AcceptPlayerSession(playerSessionID string) error
```

Parâmetros

playerSessionId

ID exclusivo emitido Amazon GameLift Servers quando uma nova sessão de jogador é criada.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

Este exemplo trata de uma solicitação de conexão que inclui a validação e a rejeição de uma sessão de jogador inválida. 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 que um jogador se desconectou do processo do servidor. Em resposta, Amazon GameLift Servers altera o slot do jogador para disponível.

Sintaxe

```
func RemovePlayerSession(playerSessionID string) error
```

Parâmetros

playerSessionId

ID exclusivo emitido Amazon GameLift Servers quando uma nova sessão de jogador é criada.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

```
err := server.RemovePlayerSession(playerSessionID)
```

DescribePlayerSessions()

Recupera dados da sessão do jogador, que incluem configurações, metadados da sessão e dados do jogador. Use esse método para obter informações sobre o seguinte:

- Uma sessão para um jogador
- Todas as sessões de jogador em uma sessão de jogo
- Todas as sessões de jogadores associadas a um único ID de jogador

Sintaxe

```
func DescribePlayerSessions(req request.DescribePlayerSessionsRequest)
    (result.DescribePlayerSessionsResult, error) {
    return srv.describePlayerSessions(&req)
}
```

Parâmetros

[DescribePlayerSessionsRequest](#)

Um objeto `DescribePlayerSessionsRequest` descreve quais sessões de jogador recuperar.

Valor de retorno

Se bem-sucedido, retorna um objeto `DescribePlayerSessionsResult` que contém um conjunto de objetos de sessão do jogador que atendem aos parâmetros de solicitação.

Exemplo

Este exemplo solicita todas as sessões de jogador conectadas ativamente a uma sessão de jogo especificada. Ao omitir `NextToken` definir o valor limite como 10, Amazon GameLift Servers retorna os primeiros registros de sessão de 10 jogadores correspondentes à solicitação.

```
// 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()

Envia uma solicitação para encontrar novos jogadores para os slots abertos em uma sessão de jogo criada com o FlexMatch. Para obter mais informações, consulte o [recurso FlexMatch de preenchimento](#).

Esta ação é assíncrona. Se novos jogadores forem combinados, Amazon GameLift Servers fornece dados atualizados do matchmaker usando a função de retorno de chamada.

`OnUpdateGameSession()`

Um processo de servidor pode ter apenas uma solicitação de alocação de correspondência ativa por vez. Para enviar uma nova solicitação, primeiro chame [StopMatchBackfill\(\)](#) para cancelar a solicitação original.

Sintaxe

```
func StartMatchBackfill(req request.StartMatchBackfillRequest)
    (result.StartMatchBackfillResult, error)
```

Parâmetros

[StartMatchBackfillRequest](#)

Um `StartMatchBackfillRequest` objeto comunica as seguintes informações:

- ID do tíquete a ser atribuído à solicitação de alocação. Essas informações são opcionais; se nenhum ID for fornecido, Amazon GameLift Servers gera um.
- O marcador de jogos para o qual a solicitação é enviada. O ARN completo da configuração é necessário. Esse valor está nos dados do marcador de jogos da sessão de jogo.
- O ID da sessão de jogo a ser preenchida.
- Os dados disponíveis de marcação para os jogadores atuais da sessão do jogo.

Valor de retorno

Retorna um objeto `StartMatchBackfillResult` com o ID do tíquete de alocação de correspondência ou um falha com uma mensagem de erro.

Exemplo

```
// 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()

Cancela uma solicitação de alocação de correspondência ativa. Para obter mais informações, consulte o [recurso FlexMatch de preenchimento](#).

Sintaxe

```
func StopMatchBackfill(req request.StopMatchBackfillRequest) error
```

Parâmetros

[StopMatchBackfillRequest](#)

Um StopMatchBackfillRequest objeto que identifica o tíquete de matchmaking a ser cancelado:

- O ID do ticket atribuído à solicitação de alocação.
- O marcador de jogo que recebeu a solicitação de alocação.
- A sessão do jogo associada à solicitação de alocação.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

```
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 o caminho para o certificado TLS usado para criptografar a conexão de rede entre o servidor do jogo e o cliente do jogo. Você pode usar o caminho do certificado ao registrar seu dispositivo computacional Amazon GameLift Servers em uma frota Anywhere. Para obter mais informações, consulte [RegisterCompute](#).

Sintaxe

```
func GetComputeCertificate() (result.GetComputeCertificateResult, error)
```

Valor de retorno

Retorna um objeto `GetComputeCertificateResult` que contém o seguinte:

- `CertificatePath`: o caminho para o certificado TLS em seu recurso computacional. Ao usar uma frota Amazon GameLift Servers gerenciada, esse caminho contém:
 - `certificate.pem`: o certificado do usuário final. A cadeia completa de certificados é a combinação de `certificateChain.pem` anexados a esse certificado.
 - `certificateChain.pem`: a cadeia de certificados que contém o certificado raiz e os certificados intermediários.
 - `rootCertificate.pem`: o certificado raiz.
 - `privateKey.pem`: a chave privada para o certificado do usuário final.
- `ComputeName`: o nome do seu recurso computacional.

Exemplo

```
tlsCertificate, err := server.GetFleetRoleCredentials(getFleetRoleCredentialsRequest)
```

GetFleetRoleCredentials()

Recupera as credenciais da função de serviço que você cria para estender as permissões à sua outra Serviços da AWS pessoa. Amazon GameLift Servers Essas credenciais permitem que seu servidor de jogo use seus recursos AWS . Para obter mais informações, consulte [Configure uma função de serviço do IAM para Amazon GameLift Servers](#).

Sintaxe

```
func GetFleetRoleCredentials(  
    req request.GetFleetRoleCredentialsRequest,  
) (result.GetFleetRoleCredentialsResult, error) {  
    return srv.getFleetRoleCredentials(&req)  
}
```

Parâmetros

[GetFleetRoleCredentialsRequest](#)

Credenciais de função que ampliam o acesso limitado aos seus AWS recursos no servidor do jogo.

Valor de retorno

Retorna um objeto `GetFleetRoleCredentialsResult` que contém o seguinte:

- `AssumedRoleUserArn` - O Amazon Resource Name (ARN) do usuário ao qual a função de serviço pertence.
- `AssumedRoleId` - O ID do usuário ao qual a função de serviço pertence.
- `AccessKeyId` - O ID da chave de acesso para autenticar e fornecer acesso aos seus AWS recursos.
- `SecretAccessKey` - O ID da chave de acesso secreta para autenticação.
- `SessionToken` - Um token para identificar a sessão ativa atual interagindo com seus AWS recursos.
- `Expiração` – A quantidade de tempo até que suas credenciais de sessão expirem.

Exemplo

```
// form the customer credentials request
getFleetRoleCredentialsRequest := request.NewGetFleetRoleCredentials()
getFleetRoleCredentialsRequest.RoleArn = "arn:aws:iam::123456789012:role/service-role/
exampleGameLiftAction"

credentials, err := server.GetFleetRoleCredentials(getFleetRoleCredentialsRequest)
```

Destroy()

Libera o SDK Amazon GameLift Servers do servidor de jogos da memória. Como melhor prática, chame esse método antes `ProcessEnding()` e depois de encerrar o processo. Se você estiver usando uma frota Anywhere e não estiver encerrando os processos do servidor após cada sessão de jogo, ligue `Destroy()` e reinicialize antes de notificar Amazon GameLift Servers que o processo está pronto para hospedar uma sessão de jogo. `InitSDK()` `ProcessReady()`

Sintaxe

```
func Destroy() error {
    return srv.destroy()
}
```

Valor de retorno

Retorna um erro com uma mensagem de erro se o método falhar.

Exemplo

```
// 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)
    }
}
```

SDK 5.x do servidor C++ (Unreal) para -- Ações Amazon GameLift Servers

Use a referência do SDK 5.x do servidor Amazon GameLift Servers Unreal para ajudá-lo a preparar seu jogo multijogador para uso com. Amazon GameLift Servers Para obter detalhes sobre o processo de integração, consulte [Adicione Amazon GameLift Servers ao seu servidor de jogos](#). Se você estiver usando o Amazon GameLift Servers plugin para Unreal, veja também [Amazon GameLift Serversplugin para Unreal Engine](#).

Note

Este tópico descreve a API Amazon GameLift Servers C++ que você pode usar ao criar para o Unreal Engine. Especificamente, essa documentação se aplica ao código que você compila com a opção `-DBUILD_FOR_UNREAL=1`.

SDK 5.x do servidor C++ (Unreal) para Amazon GameLift Servers -- Tipos de dados

Use a referência do SDK 5.x do servidor Amazon GameLift Servers Unreal para ajudá-lo a preparar seu jogo multijogador para uso com. Amazon GameLift Servers Para obter detalhes sobre o processo de integração, consulte [Adicione Amazon GameLift Servers ao seu servidor de jogos](#). Se você estiver usando o Amazon GameLift Servers plugin para Unreal, veja também [Amazon GameLift Serversplugin para Unreal Engine](#).

Note

Este tópico descreve a API Amazon GameLift Servers C++ que você pode usar ao criar para o Unreal Engine. Especificamente, essa documentação se aplica ao código que você compila com a opção `-DBUILD_FOR_UNREAL=1`.

[SDK 5.x do servidor C++ \(Unreal\) para -- Ações Amazon GameLift Servers](#)

Tipos de dados

- [FProcessParâmetros](#)
- [UpdateGameSession](#)
- [GameSession](#)
- [FServerParâmetros](#)

- [FStartMatchBackfillRequest](#)
- [FPlayer](#)
- [FGameLiftDescribePlayerSessionsRequest](#)
- [FStopMatchBackfillRequest](#)
- [FAttributeValor](#)
- [FGameLiftGetFleetRoleCredentialsRequest](#)
- [FGameLiftLongOutcome](#)
- [FGameLiftStringOutcome](#)
- [FGameLiftDescribePlayerSessionsOutcome](#)
- [FGameLiftDescribePlayerSessionsResult](#)
- [FGenericResultado](#)
- [FGameLiftPlayerSession](#)
- [FGameLiftGetComputeCertificateOutcome](#)
- [FGameLiftGetComputeCertificateResult](#)
- [FGameLiftGetFleetRoleCredentialsOutcome](#)
- [FGetFleetRoleCredentialsResult](#)
- [FGameLiftError](#)
- [Enumerações](#)

FProcessParâmetros

Esse tipo de dados contém o conjunto de parâmetros enviados Amazon GameLift Servers em [ProcessReady\(\)](#) a.

Properties	Descrição
LogParameters	Um objeto com caminhos de diretório para arquivos que são gerados durante uma sessão de jogo. Amazon GameLift Servers copia e armazena os arquivos para acesso futuro. Digite: TArray<FString> Obrigatório: não

OnHealthCheck

A função de retorno de chamada que Amazon GameLift Servers invoca para solicitar um relatório de status de saúde do processo do servidor. Amazon GameLift Servers chama essa função a cada 60 segundos e espera 60 segundos por uma resposta. O processo do servidor retorna TRUE se estiver íntegro, FALSE se não estiver íntegro. Se nenhuma resposta for retornada, Amazon GameLift Servers registra o processo do servidor como não íntegro.

Essa propriedade é uma função delegada definida como DECLARE_DELEGATE_RetVal(bool, FOnHealthCheck) ;

Digite: FOnHealthCheck

Obrigatório: não

OnProcessTerminate

A função de retorno de chamada que Amazon GameLift Servers invoca para forçar o encerramento do processo do servidor. Depois de chamar essa função, Amazon GameLift Servers espera 5 minutos até que o processo do servidor seja encerrado e responda com uma [ProcessEnding\(\)](#) chamada antes de encerrar o processo do servidor.

Digite: FSimpleDelegate

Obrigatório: Sim

OnStartGameSession

A função de retorno de chamada que Amazon GameLift Servers invoca para ativar uma nova sessão de jogo. Amazon GameLift Servers chama essa função em resposta a uma solicitação do cliente [CreateGameSession](#). A função de retorno de chamada transmite um objeto [GameSession](#).

Essa propriedade é uma função delegada definida como `DECLARE_DELEGATE_OneParam(FOnStartGameSession, Aws::GameLift::Server::Model::GameSession);`

Digite: `FOnStartGameSession`

Obrigatório: Sim

OnUpdateGameSession	A função de retorno de chamada que Amazon GameLift Servers invoca para passar um objeto de sessão de jogo atualizado para o processo do servidor. Amazon GameLift Servers chama essa função quando uma solicitação de preenchimento de correspondência é processada para fornecer dados atualizados do matchmaker. Ele transmite um objeto GameSession, uma atualização de status (updateReason) e o ID do tíquete de alocação de correspondência. Essa propriedade é uma função delegada definida como DECLARE_DELEGATE_OnParam(FOnUpdateGameSession, Aws::GameLift::Server::Model::UpdateGameSession); Digite: FOnUpdateGameSession Obrigatório: não
Porta	O número da porta em que o processo de servidor escuta para novas conexões de jogador. O valor deve estar dentro do intervalo de portas configurado para qualquer frota que implanta essa compilação do servidor de jogo. Esse número de porta está incluído nos objetos das sessões de jogo e jogador, que as sessões de jogo usam ao se conectar a um processo de servidor. Digite: int Obrigatório: Sim

UpdateGameSession

Esse tipo de dados é atualizado para um objeto de sessão de jogo, o que inclui o motivo pelo qual a sessão de jogo foi atualizada e o ID do tíquete de alocação relacionada, se a alocação for usada para preencher as sessões dos jogadores na sessão do jogo.

Propriedades	Descrição
GameSession	Um objeto GameSession. O objeto <code>GameSession</code> contém propriedades que descrevem uma sessão de jogo. Digite: <code>Aws::GameLift::Server::GameSession</code> Obrigatório: não
UpdateReason	O motivo pelo qual a sessão do jogo está sendo atualizada. Digite: <code>enum class UpdateReason</code> • <code>MATCHMAKING_DATA_UPDATED</code> • <code>BACKFILL_FAILED</code> • <code>BACKFILL_TIMED_OUT</code> • <code>BACKFILL_CANCELLED</code> Obrigatório: não
BackfillTicketId	O ID do tíquete de alocação que está tentando atualizar a sessão do jogo. Digite: <code>char[]</code> Obrigatório: não

GameSession

Esse tipo de dados fornece detalhes de uma sessão de jogo.

Propriedades	Descrição
GameSessionId	Um identificador exclusivo da sessão de jogo. Um ARN de sessão de jogo tem o seguinte formato: <code>arn:aws:gamelift:<region>::gamesession/<fleet ID>/<custom ID string or idempotency token></code> . Digite: <code>char[]</code> Obrigatório: não
Name	Um rótulo descritivo da sessão do jogo. Digite: <code>char[]</code> Obrigatório: não
FleetId	Um identificador exclusivo da frota em que a sessão de jogo está sendo executada. Digite: <code>char[]</code> Obrigatório: não
MaximumPlayerSessionCount	O número máximo de conexões do jogador com a sessão do jogo. Digite: <code>int</code> Obrigatório: não
Porta	O número da porta da sessão de jogo. Para se conectar a um servidor de Amazon GameLift Servers jogos, um aplicativo precisa do endereço IP e do número da porta. Digite: <code>int</code>

Propriedades	Descrição
	Obrigatório: não
IpAddress	O endereço IP da sessão de jogo. Para se conectar a um servidor de Amazon GameLift Servers jogos, um aplicativo precisa do endereço IP e do número da porta. Digite: <code>char[]</code> Obrigatório: não
GameSessionData	Um conjunto de propriedades de sessão de jogo personalizadas, formatadas como um único valor de string. Digite: <code>char[]</code> Obrigatório: não
MatchmakerData	Informações sobre o processo de matchmaking usado para criar a sessão do jogo, na sintaxe JSON, formatado como uma string. Além da configuração de marcação de jogos usada, ele contém dados sobre todos os jogadores atribuídos à partida, incluindo atributos do jogador e atribuições da equipe. Digite: <code>char[]</code> Obrigatório: não

Propriedades	Descrição
GameProperties	Um conjunto de propriedades personalizadas para uma sessão de jogo, formatado como pares de chave:valor. Essas propriedades são passadas com uma solicitação de início de uma nova sessão de jogo. Digite: <code>GameProperty[]</code> Obrigatório: não
DnsName	O identificador DNS atribuído à instância que está executando a sessão do jogo. Os valores têm o formato a seguir: - Frotas habilitadas para TLS: <code><unique identifier>.<region identifier>.amazongamelift.com</code> . - Non-TLS-enabled frotas: <code>ec2-<unique identifier>.compute.amazonaws.com</code> . Ao se conectar a uma sessão de jogo que está sendo executada em uma frota compatível com TLS, você deve usar o nome DNS, não o endereço IP. Digite: <code>char[]</code> Obrigatório: não

FServerParâmetros

Informações usadas para manter a conexão entre um servidor Amazon GameLift Servers Anywhere e o Amazon GameLift Servers serviço. Essas informações são usadas ao iniciar novos processos de servidor com [InitSDK\(\)](#). Para servidores hospedados em EC2 instâncias Amazon GameLift Servers gerenciadas, use um objeto vazio.

Propriedades	Descrição
webSocketUrl	Os <code>GameLiftServerSdkEndpoint</code> Amazon GameLift Servers retornos quando você RegisterCompute compra um recurso de computação Amazon GameLift Servers Anywhere. Digite: <code>char[]</code> Obrigatório: Sim
processId	Um identificador exclusivo registrado no processo do servidor que hospeda seu jogo. Digite: <code>char[]</code> Obrigatório: Sim
hostId	O <code>HostID</code> é o <code>ComputeName</code> usado quando você registrou seu computador. Para obter mais informações, consulte, RegisterCompute. Digite: <code>char[]</code> Obrigatório: Sim
fleetId	O identificador exclusivo da frota na qual o computador está registrado. Para obter mais informações, consulte, RegisterCompute. Digite: <code>char[]</code> Obrigatório: Sim
authToken	O token de autenticação gerado por Amazon GameLift Servers isso autentica seu servidor em Amazon GameLift Servers. Para obter mais informações, consulte, GetComputeAuthToken.

Propriedades	Descrição
	Digite: <code>char[]</code>
	Obrigatório: Sim

FStartMatchBackfillRequest

Informações usadas para criar uma solicitação de alocação de marcação de jogos. O servidor do jogo comunica essas informações Amazon GameLift Servers em uma [StartMatchBackfill\(\)](#) chamada.

Propriedades	Descrição
GameSessionArn	Um identificador da sessão de jogo exclusivo . A operação de API retorna GetGameSessionId o identificador no formato de ARN. Digite: <code>char[]</code> Obrigatório: Sim
MatchmakingConfigurationArn	um identificador exclusivo, no formato de ARN, para o marcador de jogos a ser usado para essa solicitação. O marcador de jogos ARN da sessão de jogo original está no objeto de sessão de jogo na propriedade de dados do marcador de jogos. Saiba mais sobre os dados do marcador de jogos em Trabalhar com dados do marcador de jogos . Digite: <code>char[]</code> Obrigatório: Sim
Jogadores	Um conjunto de dados que representa todos os jogadores que estão na sessão do jogo. O marcador de jogos usa essas informações para pesquisar novos jogadores que sejam

Propriedades	Descrição
	correspondências ideais para os jogadores atuais. Digite: TArray<FPlayer> Obrigatório: Sim
TicketId	Um identificador exclusivo de um tíquete de solicitação de marcação de jogo ou de alocação de correspondência. Se você não fornecer um valor, Amazon GameLift Servers gera um. Use esse identificador para acompanhar o status do tíquete de alocação de correspondência ou cancelar a solicitação, se necessário. Digite: char[] Obrigatório: não

FPlayer

Esse tipo de dados representa um jogador em marcação de jogos. Ao iniciar uma solicitação de matchmaking, um jogador tem um ID de jogador, atributos, e possivelmente dados de latência. Amazon GameLift Servers adiciona informações da equipe após a realização de uma partida.

Propriedades	Descrição
LatencyInMS	Um conjunto de valores expressos em milissegundos que indicam a quantidade de latência que um jogador experimenta quando conectado a um local. Se essa propriedade for usada, o player só será correspondido aos locais listados. Se um marcador de jogos tiver uma regra que avalie

Propriedades	Descrição
	a latência do jogador, os jogadores deverão reportar a latência a ser correspondida. Digite: <code>TMap>FString, int32<</code> Obrigatório: não
PlayerAttributes	Uma coleção de pares de chave:valor contendo informações de jogadores para uso em marcação de jogos. As chaves de atributos do jogador devem corresponder às PlayerAttributes usadas em um conjunto de regras de matchmaking. Para obter mais informações sobre os atributos do jogador, consulte AttributeValue. Digite: <code>TMap>FString, FAttributeValue<</code> Obrigatório: não
PlayerId	Um identificador exclusivo para um jogador. Digite: <code>std::string</code> Obrigatório: não
Equipe	O nome da equipe à qual o jogador está designado em uma partida. Você define o nome da equipe no conjunto de regras de marcação de jogos. Digite: <code>FString</code> Obrigatório: não

FGameLiftDescribePlayerSessionsRequest

Um objeto que especifica quais sessões de jogador devem ser recuperadas. O processo do servidor fornece essas informações com uma [DescribePlayerSessions\(\)](#) chamada para Amazon GameLift Servers.

Propriedades	Descrição
GameSessionId	Um identificador da sessão de jogo exclusivo . Use esse parâmetro para solicitar todas as sessões de jogador da sessão de jogo especificada. O formato do ID da sessão do jogo é FString. O GameSessionID é uma string de ID personalizada ou uma Digite: std::string Obrigatório: não
PlayerSessionId	Um identificador exclusivo para uma sessão de jogador. Use esse parâmetro para solicitar uma única sessão de jogador específica. Digite: FString Obrigatório: não
PlayerId	Um identificador exclusivo para um jogador. Use este parâmetro para solicitar todas as sessões de jogador para um jogador específico. Consulte Gerar jogador IDs. Digite: FString Obrigatório: não

Propriedades	Descrição
PlayerSessionStatusFilter	O status da sessão de jogador para filtrar resultados. Entre os status da sessão de jogador possíveis incluem: • RESERVED - A solicitação de sessão do jogador foi recebida, mas o jogador não se conectou ao processo do servidor nem foi validado. • ACTIVE - O jogador foi validado pelo processo do servidor e está conectado. • COMPLETED – A conexão do jogador foi interrompida. • TIMEDOUT – Uma solicitação de sessão de jogador foi recebida, mas o jogador não se conectou ou não foi validado dentro do tempo limite (60 segundos). Digite: FString Obrigatório: não
NextToken	O token que indica o início da próxima página de resultados. Para especificar o início do conjunto de resultados, não forneça um valor. Se fornecer um ID de sessão do jogador, esse parâmetro será ignorado. Digite: FString Obrigatório: não

Propriedades	Descrição
Limite	O número máximo de resultados a serem retornados. Se fornecer um ID de sessão do jogador, esse parâmetro será ignorado. Digite: <code>int</code> Obrigatório: não

FStopMatchBackfillRequest

Informações usadas para cancelar uma solicitação de alocação de marcação de jogos. O servidor do jogo comunica essas informações ao Amazon GameLift Servers serviço em uma [StopMatchBackfill\(\)](#) chamada.

Propriedades	Descrição
GameSessionArn	Um identificador exclusivo da sessão de jogo da solicitação que está sendo cancelada. Digite: <code>FString</code> Obrigatório: Sim
MatchmakingConfigurationArn	Um identificador exclusivo do marcador de jogos para o qual essa solicitação foi enviada. Digite: <code>FString</code> Obrigatório: Sim
TicketId	Um identificador exclusivo do tíquete de solicitação de alocação a ser cancelado. Digite: <code>FString</code> Obrigatório: Sim

FAttributeValor

Use esses valores em pares de chave-valor de atributo [FPlayer](#). Esse objeto permite especificar um valor de atributo usando qualquer um dos tipos de dados válidos: string, número, matriz de string ou mapa de dados. Cada objeto `AttributeValue` pode usar somente uma das propriedades disponíveis.

Propriedades	Descrição
<code>attrType</code>	Especifica o tipo de valor do atributo. Tipo: Um <code>FAttributeType</code> valor enum. Obrigatório: não
<code>S</code>	Representa um valor de atributo de string. Digite: <code>FString</code> Obrigatório: não
<code>N</code>	Representa um valor de atributo numérico. Digite: <code>double</code> Obrigatório: não
<code>SL</code>	Representa uma matriz de valores de atributos de string. Digite: <code>TArray<FString></code> Obrigatório: não
<code>SDM</code>	Representa um dicionário de chaves de string e valores duplos. Digite: <code>TMap<FString, double></code> Obrigatório: não

FGameLiftGetFleetRoleCredentialsRequest

Esse tipo de dados fornece credenciais de função que ampliam o acesso limitado aos seus recursos AWS no servidor do jogo. Para mais informações, consulte [Configure uma função de serviço do IAM para Amazon GameLift Servers](#).

Propriedades	Descrição
RoleArn	O Amazon Resource Name (ARN) da função de serviço que estende o acesso limitado aos seus AWS recursos. Digite: FString Obrigatório: não
RoleSessionName	O nome da sessão que descreve o uso das credenciais da função. Digite: FString Obrigatório: não

FGameLiftLongOutcome

Esse tipo de dados resulta de uma ação e produz um objeto com as seguintes propriedades:

Propriedades	Descrição
Resultado	O resultado da ação. Digite: long Obrigatório: não
ResultWithOwnership	O resultado da ação, convertido como um rvalue, para que o código de chamada possa se apropriar do objeto. Digite: long&&

Propriedades	Descrição
	Obrigatório: não
Bem-sucedida	Se a ação foi bem-sucedida ou não. Digite: <code>bool</code> Obrigatório: Sim
Erro	O erro que ocorreu se a ação não foi bem-sucedida. Digite: the section called “FGameLiftError” Obrigatório: não

FGameLiftStringOutcome

Esse tipo de dados resulta de uma ação e produz um objeto com as seguintes propriedades:

Propriedades	Descrição
Resultado	O resultado da ação. Digite: <code>FString</code> Obrigatório: não
ResultWithOwnership	O resultado da ação, convertido como um rvalue, para que o código de chamada possa se apropriar do objeto. Digite: <code>FString&&</code> Obrigatório: não
Bem-sucedida	Se a ação foi bem-sucedida ou não. Digite: <code>bool</code> Obrigatório: Sim
Erro	O erro que ocorreu se a ação não foi bem-sucedida.

Propriedades	Descrição
	Digite: the section called “FGameLiftError”
	Obrigatório: não

FGameLiftDescribePlayerSessionsOutcome

Esse tipo de dados resulta de uma ação e produz um objeto com as seguintes propriedades:

Propriedades	Descrição
Resultado	O resultado da ação. Digite: the section called “FGameLiftDescribePlayerSessionsResult” Obrigatório: não
ResultWithOwnership	O resultado da ação, convertido como um rvalue, para que o código de chamada possa se apropriar do objeto. Digite: <code>FGameLiftDescribePlayerSessionsResult&&</code> Obrigatório: não
Bem-sucedida	Se a ação foi bem-sucedida ou não. Digite: <code>bool</code> Obrigatório: Sim
Erro	O erro que ocorreu se a ação não foi bem-sucedida. Digite: the section called “FGameLiftError” Obrigatório: não

FGameLiftDescribePlayerSessionsResult

Propriedades	Descrição
PlayerSessions	Digite: TArray<FGameLiftPlayerSession> Obrigatório: Sim
NextToken	O token indicando o início da próxima página de resultados. Para especificar o início do conjunto de resultados, não forneça um valor. Se fornecer um ID de sessão do jogador, esse parâmetro será ignorado. Digite: FString Obrigatório: não
Bem-sucedida	Se a ação foi bem-sucedida ou não. Digite: bool Obrigatório: Sim
Erro	O erro que ocorreu se a ação não foi bem-sucedida. Digite: the section called “FGameLiftError” Obrigatório: não

FGenericResultado

Esse tipo de dados resulta de uma ação e produz um objeto com as seguintes propriedades:

Propriedades	Descrição
Bem-sucedida	Se a ação foi bem-sucedida ou não. Digite: bool

Propriedades	Descrição
	Obrigatório: Sim
Erro	O erro que ocorreu se a ação não foi bem-sucedida. Digite: the section called “FGameLiftError” Obrigatório: não

FGameLiftPlayerSession

Propriedades	Descrição
CreationTime	Digite: long Obrigatório: Sim
FleetId	Digite: FString Obrigatório: Sim
GameSessionId	Digite: FString Obrigatório: Sim
IpAddress	Digite: FString Obrigatório: Sim
PlayerData	Digite: FString Obrigatório: Sim
PlayerId	Digite: FString Obrigatório: Sim
PlayerSessionId	Digite: FString Obrigatório: Sim

Propriedades	Descrição
Porta	Digite: <code>int</code> Obrigatório: Sim
Status	Tipo: Um <code>PlayerSessionStatus</code> enum.. Obrigatório: Sim
TerminationTime	Digite: <code>long</code> Obrigatório: Sim
DnsName	Digite: <code>FString</code> Obrigatório: Sim

FGameLiftGetComputeCertificateOutcome

Esse tipo de dados resulta de uma ação e produz um objeto com as seguintes propriedades:

Propriedades	Descrição
Resultado	O resultado da ação. Digite: the section called “FGameLiftGetComputeCertificateResult” Obrigatório: não
ResultWithOwnership	O resultado da ação, convertido como um rvalue, para que o código de chamada possa se apropriar do objeto. Digite: <code>FGameLiftGetComputeCertificateResult&&</code> Obrigatório: não
Bem-sucedida	Se a ação foi bem-sucedida ou não. Digite: <code>bool</code>

Propriedades	Descrição
	Obrigatório: Sim
Erro	O erro que ocorreu se a ação não foi bem-sucedida. Digite: the section called “FGameLiftError” Obrigatório: não

FGameLiftGetComputeCertificateResult

O caminho para o certificado TLS em seu computador e o nome do host do computador.

Propriedades	Descrição
CertificatePath	Digite: FString Obrigatório: Sim
ComputeName	Digite: FString Obrigatório: Sim

FGameLiftGetFleetRoleCredentialsOutcome

Esse tipo de dados resulta de uma ação e produz um objeto com as seguintes propriedades:

Propriedades	Descrição
Resultado	O resultado da ação. Digite: the section called “FGetFleetRoleCredentialsResult” Obrigatório: não
ResultWithOwnership	O resultado da ação, convertido como um rvalue, para que o código de chamada possa se apropriar do objeto.

Propriedades	Descrição
	Digite: <code>FGameLiftGetFleetRoleCredentialsResult</code> Obrigatório: não
Bem-sucedida	Se a ação foi bem-sucedida ou não. Digite: <code>bool</code> Obrigatório: Sim
Erro	O erro que ocorreu se a ação não foi bem-sucedida. Digite: the section called “FGameLiftError” Obrigatório: não

FGetFleetRoleCredentialsResult

Propriedades	Descrição
AccessKeyId	O ID da chave de acesso para autenticar e fornecer acesso aos seus recursos AWS . Digite: <code>FString</code> Obrigatório: não
AssumedRoleId	O ID do usuário ao qual o perfil de serviço pertence. Digite: <code>FString</code> Obrigatório: não
AssumedRoleUserArn	O nome do recurso da Amazon (ARN) do usuário ao qual pertence o perfil de serviço. Digite: <code>FString</code>

Propriedades	Descrição
	Obrigatório: não
Expiração	A quantidade de tempo até que suas credenciais de sessão expirem. Digite: <code>FDateTime</code> Obrigatório: não
SecretAccessKey	O ID da chave de acesso secreta para autenticação. Digite: <code>FString</code> Obrigatório: não
SessionToken	Um token para identificar a sessão ativa atual interagindo com seus AWS recursos. Digite: <code>FString</code> Obrigatório: não
Bem-sucedida	Se a ação foi bem-sucedida ou não. Digite: <code>bool</code> Obrigatório: Sim
Erro	O erro que ocorreu se a ação não foi bem-sucedida. Digite: the section called “GameLiftError” Obrigatório: não

FGameLiftError

Propriedades	Descrição
ErrorType	O tipo de erro.

Propriedades	Descrição
	Tipo: Um <code>GameLiftErrorType</code> enum .. Obrigatório: não
ErrorName	O nome do erro. Digite: <code>std::string</code> Obrigatório: não
ErrorMessage	A mensagem de erro. Digite: <code>std::string</code> Obrigatório: não

Enumerações

Os enums definidos para o SDK do servidor para Amazon GameLift Servers (Unreal) são definidos da seguinte forma:

FAttributeTipo

- NONE
- STRING
- DOUBLE
- STRING_LIST
- STRING_DOUBLE_MAP

GameLiftErrorType

Valor da string indicando o tipo de erro. Os valores válidos são:

- SERVICE_CALL_FAILED — Uma chamada para um serviço falhou. AWS
- LOCAL_CONNECTION_FAILED — A conexão local falhou. Amazon GameLift Servers
- NETWORK_NOT_INITIALIZED – A rede não foi inicializada.
- GAMESESSION_ID_NOT_SET – O ID da sessão do jogo não foi definido.
- BAD_REQUEST_EXCEPTION

- `INTERNAL_SERVICE_EXCEPTION`
- `ALREADY_INITIALIZED` — O Amazon GameLift Servers servidor ou cliente já foi inicializado com `Initialize ()`.
- `FLEET_MISMATCH` – A frota de destino não corresponde à frota de uma `GameSession` ou `PlayerSession`.
- `GAMELIFT_CLIENT_NOT_INITIALIZED` — O cliente não foi inicializado. Amazon GameLift Servers
- `GAMELIFT_SERVER_NOT_INITIALIZED` — O servidor não foi inicializado. Amazon GameLift Servers
- `GAME_SESSION_ENDED_FAILED` — o SDK do servidor não conseguiu entrar em contato com o serviço para informar Amazon GameLift Servers que a sessão do jogo foi encerrada.
- `GAME_SESSION_NOT_READY` — A sessão de jogo do servidor não foi ativada Amazon GameLift Servers.
- `GAME_SESSION_READY_FAILED` — o SDK do servidor não conseguiu entrar em contato com o serviço para informar Amazon GameLift Servers que a sessão do jogo está pronta.
- `INITIALIZATION_MISMATCH` – Um método de cliente foi chamado depois de `Server::Initialize()`, ou vice-versa.
- `NOT_INITIALIZED` — O Amazon GameLift Servers servidor ou cliente não foi inicializado com `Initialize ()`.
- `NO_TARGET_ALIASID_SET` – Um `aliasID` de destino não foi definido.
- `NO_TARGET_FLEET_SET` – Uma frota alvo não foi definida.
- `PROCESS_ENDING_FAILED` — o SDK do servidor não conseguiu entrar em contato com o serviço para Amazon GameLift Servers informar que o processo está sendo encerrado.
- `PROCESS_NOT_ACTIVE` — O processo do servidor ainda não está ativo, não está vinculado a um e não pode `GameSession` aceitar ou processar. `PlayerSessions`
- `PROCESS_NOT_READY` – O processo do servidor ainda não está pronto para ser ativado.
- `PROCESS_READY_FAILED` — o SDK do servidor não conseguiu entrar em contato com o serviço para informar que o processo está pronto.
- `SDK_VERSION_DETECTION_FAILED` – Falha na detecção da versão do SDK.
- `STX_CALL_FAILED` — Uma chamada para o componente de back-end do XStx servidor falhou.
- `STX_INITIALIZATION_FAILED` — O componente de back-end do XStx servidor falhou ao inicializar.

- UNEXPECTED_PLAYER_SESSION – Uma sessão de jogador não registrada foi encontrada pelo servidor.
- WEBSOCKET_CONNECT_FAILURE
- WEBSOCKET_CONNECT_FAILURE_FORBIDDEN
- WEBSOCKET_CONNECT_FAILURE_INVALID_URL
- WEBSOCKET_CONNECT_FAILURE_TIMEOUT
- WEBSOCKET_RETRIABLE_SEND_MESSAGE_FAILURE — Falha recuperável ao enviar uma mensagem ao Serviço. GameLift WebSocket
- WEBSOCKET_SEND_MESSAGE_FAILURE — Falha ao enviar uma mensagem para o serviço. GameLift WebSocket
- MATCH_BACKFILL_REQUEST_VALIDATION – A validação da solicitação falhou.
- PLAYER_SESSION_REQUEST_VALIDATION – A validação da solicitação falhou.

EPlayerSessionCreationPolicy

Valor de string que indica se a sessão do jogo aceita novos jogadores. Os valores válidos são:

- ACCEPT_ALL – Aceite todas as novas sessões de jogador.
- DENY_ALL – Recuse todas as novas sessões de jogador.
- NOT_SET – A sessão do jogo não está configurada para aceitar ou negar sessões de novos jogadores.

EPlayerSessionStatus

- ACTIVE
- COMPLETED
- NOT_SET
- RESERVED
- TIMEDOUT

[SDK 5.x do servidor C++ \(Unreal\) para Amazon GameLift Servers -- Tipos de dados](#)

Tópicos

- [GetSdkVersion\(\)](#)
- [InitSDK\(\)](#)
- [InitSDK\(\)](#)
- [ProcessReady\(\)](#)

- [ProcessEnding\(\)](#)
- [ActivateGameSession\(\)](#)
- [UpdatePlayerSessionCreationPolicy\(\)](#)
- [GetGameSessionId\(\)](#)
- [GetTerminationTime\(\)](#)
- [AcceptPlayerSession\(\)](#)
- [RemovePlayerSession\(\)](#)
- [DescribePlayerSessions\(\)](#)
- [StartMatchBackfill\(\)](#)
- [StopMatchBackfill\(\)](#)
- [GetComputeCertificate\(\)](#)
- [GetFleetRoleCredentials\(\)](#)

GetSdkVersion()

Retorna o número da versão atual do SDK compilado no processo de servidor.

Sintaxe

```
FGameLiftStringOutcome GetSdkVersion();
```

Valor de retorno

Se bem-sucedido, retornará a versão do SDK atual como um objeto [the section called “FGameLiftStringOutcome”](#). O objeto retornada inclui apenas o número da versão (exemplo, 5.0.0). Se não for bem-sucedido, retornará uma mensagem de erro.

Exemplo

```
Aws::GameLift::AwsStringOutcome SdkVersionOutcome =  
    Aws::GameLift::Server::GetSdkVersion();
```

InitSDK()

Inicializa o Amazon GameLift Servers SDK para uma frota gerenciada EC2 . Chame esse método na inicialização, antes que qualquer outra inicialização relacionada Amazon GameLift Servers ocorra.

Esse método lê os parâmetros do servidor do ambiente host para configurar a comunicação entre o servidor e o Amazon GameLift Servers serviço.

Sintaxe

```
FGameLiftGenericOutcome InitSDK()
```

Valor de retorno

Se bem-sucedido, retornará um objeto `InitSdkOutcome` indicando que o processo de servidor está pronto para chamar [ProcessReady\(\)](#).

Exemplo

```
//Call InitSDK to establish a local connection with the Amazon GameLift Servers Agent  
to enable further communication.  
FGameLiftGenericOutcome initSdkOutcome = gameLiftSdkModule->InitSDK();
```

InitSDK()

Inicializa o Amazon GameLift Servers SDK para uma frota Anywhere ou uma frota de contêineres gerenciados. Chame esse método na inicialização, antes que qualquer outra inicialização relacionada Amazon GameLift Servers ocorra. Esse método requer parâmetros explícitos do servidor para configurar a comunicação entre o servidor e o Amazon GameLift Servers serviço.

Sintaxe

```
FGameLiftGenericOutcome InitSDK(serverParameters)
```

Parâmetros

[FServerParâmetros](#)

Para inicializar um servidor de jogo em uma frota Amazon GameLift Servers Anywhere, construa um `ServerParameters` objeto com as seguintes informações:

- O URL do WebSocket usado para se conectar ao seu servidor de jogo.
- O ID do processo usado para hospedar o servidor de jogos.
- O ID do computador que hospeda os processos do seu servidor de jogos.

- O ID da Amazon GameLift Servers frota que contém sua computação Amazon GameLift Servers Anywhere.
- O token de autorização gerado pela Amazon GameLift Servers operação.

Valor de retorno

Se bem-sucedido, retornará um objeto `InitSdkOutcome` indicando que o processo de servidor está pronto para chamar [ProcessReady\(\)](#).

Note

Se as chamadas para `InitSDK()` estiverem falhando para compilações de jogos implantadas em frotas do Anywhere, verifique o parâmetro `ServerSdkVersion` usado ao criar o recurso de compilação. Você deve definir explicitamente esse valor para a versão do SDK do servidor em uso. O valor padrão desse parâmetro é 4.x, o que não é compatível. Para resolver esse problema, crie uma nova versão e implante-a em uma nova frota.

Exemplo

```
//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 que o processo do servidor está pronto para hospedar sessões de jogo. Chame esse método após invocar [InitSDK\(\)](#). Esse método deve ser chamado somente uma vez por processo.

Sintaxe

```
GenericOutcome ProcessReady(const Aws::GameLift::Server::ProcessParameters  
&processParameters);
```

Parâmetros

processParameters

Um objeto [FProcessParâmetros](#) que fornece as seguintes informações sobre o processo do servidor:

- Nomes dos métodos de retorno de chamada implementados no código do servidor do jogo que o Amazon GameLift Servers serviço invoca para se comunicar com o processo do servidor.
- Número da porta em que o processo de servidor está escutando.
- Caminho de qualquer arquivo específico da sessão do jogo que você deseja que o Amazon GameLift Servers capture e armazene.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

Este exemplo ilustra as implementações das funções de chamada e delegação [ProcessReady\(\)](#).

```
//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 que o processo do servidor está sendo encerrado. Chame esse método depois de todas as outras tarefas de limpeza (incluindo o encerramento da sessão ativa do jogo) e antes de encerrar o processo. Dependendo do resultado `ProcessEnding()`, o processo sai com sucesso (0) ou erro (-1) e gera um evento de frota. Se o processo for encerrado com um erro, o evento de frota gerado será `SERVER_PROCESS_TERMINATED_UNHEALTHY`.

Sintaxe

```
FGameLiftGenericOutcome ProcessEnding()
```

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

```
//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 que o processo do servidor ativou uma sessão de jogo e agora está pronto para receber conexões de jogadores. Essa ação deve ser chamada como parte da função de retorno de chamada `onStartGameSession()`, após toda inicialização da sessão do jogo.

Sintaxe

```
FGameLiftGenericOutcome ActivateGameSession()
```

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

Este exemplo mostra chamado `ActivateGameSession()` como parte da função de delegação `onStartGameSession()`.

```
//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()

Atualiza a capacidade da sessão do jogo atual para aceitar novas sessões de jogador. Atualiza a capacidade da sessão do jogo atual para aceitar novas sessões de jogador.

Sintaxe

```
FGameLiftGenericOutcome UpdatePlayerSessionCreationPolicy(EPlayerSessionCreationPolicy
policy)
```

Parâmetros

playerCreationSessionPolítica

Valor de string que indica se a sessão do jogo aceita novos jogadores.

Os valores válidos são:

- ACCEPT_ALL – Aceite todas as novas sessões de jogador.
- DENY_ALL – Recuse todas as novas sessões de jogador.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

Este exemplo define a política de ingresso da sessão do jogo atual para aceitar todos os jogadores.

```
FGameLiftGenericOutcome outcome = gameLiftSdkModule-  
>UpdatePlayerSessionCreationPolicy(Aws::GameLift::Model::EPlayerSessionCreationPolicy::ACCEPT_A
```

GetGameSessionId()

Recupera o ID da sessão de jogo hospedada pelo processo do servidor ativo.

Para processos inativos que não são ativados com uma sessão de jogo, a chamada retorna um [the section called “FGameLiftError”](#).

Sintaxe

```
FGameLiftStringOutcome GetGameSessionId()
```

Parâmetros

Essa ação não tem um parâmetro.

Valor de retorno

Se bem-sucedido, retornará o ID da sessão de jogo como um objeto [the section called “FGameLiftStringOutcome”](#). Se não for bem-sucedido, retornará uma mensagem de erro."

Para processos inativos que não são ativados com uma sessão de jogo, a chamada retorna Success=True e GameSessionId="".

Exemplo

```
//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();  
};
```

GetTerminationTime()

Retorna a hora em que um processo do servidor está programado para ser desligado, se essa informação estiver disponível. Um processo do servidor age após receber um `onProcessTerminate()` retorno de chamada de Amazon GameLift Servers. Amazon GameLift Servers chamadas `onProcessTerminate()` pelos seguintes motivos:

- Quando o processo do servidor relatou problemas de saúde ou não respondeu Amazon GameLift Servers.
- Ao encerrar a instância durante um evento de redução.
- Quando uma instância é encerrada devido a uma [interrupção na instância spot](#).

Sintaxe

```
AwsDateTimeOutcome GetTerminationTime()
```

Valor de retorno

Se for bem-sucedido, retornará o horário de término como um objeto `AwsDateTimeOutcome`. O valor é o tempo de rescisão, expresso em tiques decorridos desde então `0001 00:00:00`. Por exemplo, o valor da data e hora `2020-09-13 12:26:40 -000Z` é igual aos tiques `637355968000000000`. Se nenhum horário de encerramento estiver disponível, o retornará uma mensagem de erro.

Se o processo não tiver recebido um retorno de chamada `ProcessParameters.OnProcessTerminate()`, uma mensagem de erro será retornada. Para obter mais informações sobre como desligar um processo do servidor, consulte [Responder a uma notificação de desligamento do processo do servidor](#).

Exemplo

```
AwsDateTimeOutcome TermTimeOutcome = gameLiftSdkModule->GetTerminationTime();
```

AcceptPlayerSession()

Notifica Amazon GameLift Servers que um jogador com o ID de sessão de jogador especificado se conectou ao processo do servidor e precisa de validação. Amazon GameLift Servers verifica se o ID

da sessão do jogador é válido. Depois que a sessão do jogador for validada, o status da vaga do jogador é Amazon GameLift Servers alterado de RESERVADO para ATIVO.

Sintaxe

```
FGameLiftGenericOutcome AcceptPlayerSession(const FString& playerId)
```

Parâmetros

playerSessionId

ID exclusivo emitido Amazon GameLift Servers quando uma nova sessão de jogador é criada.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

Este exemplo trata de uma solicitação de conexão que inclui a validação e a rejeição de uma sessão de jogador inválida. IDs

```
bool GameLiftManager::AcceptPlayerSession(const FString& playerId, const
FString& gameId)
{
    #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 que um jogador se desconectou do processo do servidor. Em resposta, Amazon GameLift Servers altera o slot do jogador para disponível.

Sintaxe

```
FGameLiftGenericOutcome RemovePlayerSession(const FString& playerSessionId)
```

Parâmetros

playerSessionId

ID exclusivo emitido Amazon GameLift Servers quando uma nova sessão de jogador é criada.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

```
bool GameLiftManager::RemovePlayerSession(const FString& playerSessionId)
{
    #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 dados da sessão do jogador, que incluem configurações, metadados da sessão e dados do jogador. Use esse método para obter informações sobre o seguinte:

- Uma sessão para um jogador
- Todas as sessões de jogador em uma sessão de jogo
- Todas as sessões de jogadores associadas a um único ID de jogador

Sintaxe

```
FGameLiftDescribePlayerSessionsOutcome DescribePlayerSessions(const
FGameLiftDescribePlayerSessionsRequest &describePlayerSessionsRequest)
```

Parâmetros

[FGameLiftDescribePlayerSessionsRequest](#)

Um objeto [the section called “FGameLiftDescribePlayerSessionsRequest”](#) que descreve quais sessões de jogador recuperar.

Valor de retorno

Se bem-sucedido, retorna um objeto [the section called “FGameLiftDescribePlayerSessionsOutcome”](#) que contém um conjunto de objetos de sessão do jogador que atendem aos parâmetros de solicitação.

Exemplo

Este exemplo solicita todas as sessões de jogador conectadas ativamente a uma sessão de jogo especificada. Ao omitir NextToken e definir o valor limite como 10, Amazon GameLift Servers retorna os primeiros registros de sessão de 10 jogadores que correspondam à solicitação.

```
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()

Envia uma solicitação para encontrar novos jogadores para os slots abertos em uma sessão de jogo criada com o FlexMatch. Para obter mais informações, consulte o [recurso FlexMatch de preenchimento](#).

Esta ação é assíncrona. Se novos jogadores forem combinados, Amazon GameLift Servers fornece dados atualizados do matchmaker usando a função de retorno de chamada.

OnUpdateGameSession()

Um processo de servidor pode ter apenas uma solicitação de alocação de correspondência ativa por vez. Para enviar uma nova solicitação, primeiro chame [StopMatchBackfill\(\)](#) para cancelar a solicitação original.

Sintaxe

```
FGameLiftStringOutcome StartMatchBackfill (FStartMatchBackfillRequest  
&startBackfillRequest);
```

Parâmetros

[FStartMatchBackfillRequest](#)

Um `StartMatchBackfillRequest` objeto que comunica as seguintes informações:

- ID do tíquete a ser atribuído à solicitação de alocação. Essas informações são opcionais; se nenhum ID for fornecido, Amazon GameLift Servers será gerado um.
- O marcador de jogos para o qual a solicitação é enviada. O ARN completo da configuração é necessário. Esse valor está nos dados do marcador de jogos da sessão de jogo.
- O ID da sessão de jogo a ser preenchida.
- Os dados disponíveis de marcação para os jogadores atuais da sessão do jogo.

Valor de retorno

Retorna um objeto `StartMatchBackfillOutcome` com o ID do tíquete de alocação de correspondência ou um falha com uma mensagem de erro.

Exemplo

```
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()

Cancela uma solicitação de alocação de correspondência ativa. Para obter mais informações, consulte o [recurso FlexMatch de preenchimento](#).

Sintaxe

```
FGameLiftGenericOutcome StopMatchBackfill (FStopMatchBackfillRequest  
&stopBackfillRequest);
```

Parâmetros

FStopMatchBackfillRequest

Um StopMatchBackfillRequest objeto identificando o tíquete de matchmaking a ser cancelado:

- O ID do ticket atribuído à solicitação de alocação.
- O marcador de jogo que recebeu a solicitação de alocação.
- A sessão do jogo associada à solicitação de alocação.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

```
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 o caminho para o certificado TLS usado para criptografar a conexão de rede entre seu recurso computacional Amazon GameLift Servers Anywhere e. Amazon GameLift Servers Você pode usar o caminho do certificado ao registrar seu dispositivo computacional Amazon GameLift Servers em uma frota Anywhere. Para obter mais informações, consulte, [RegisterCompute](#).

Sintaxe

```
FGameLiftGetComputeCertificateOutcome FGameLiftServerSDKModule::GetComputeCertificate()
```

Valor de retorno

Retorna um objeto GetComputeCertificateResponse contendo o seguinte:

- CertificatePath: o caminho para o certificado TLS em seu recurso computacional.
- HostName: o nome do host do seu recurso computacional.

Exemplo

```

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_T0_TCHAR(outres.GetCertificatePath());
        result.m_computeName = UTF8_T0_TCHAR(outres.GetComputeName());
        return FGameLiftGetComputeCertificateOutcome(result);
    }
    else {
        return FGameLiftGetComputeCertificateOutcome(FGameLiftError(outcome.GetError()));
    }
}

```

```

}
#else
return FGameLiftGetComputeCertificateOutcome(FGameLiftGetComputeCertificateResult());
#endif
}

```

GetFleetRoleCredentials()

Recupera as credenciais da função do IAM que autorizam Amazon GameLift Servers a interação com outras pessoas. Serviços da AWS Para obter mais informações, consulte [Comunique-se com outros AWS recursos de suas frotas](#).

Sintaxe

```

FGameLiftGetFleetRoleCredentialsOutcome
FGameLiftServerSDKModule::GetFleetRoleCredentials(const
FGameLiftGetFleetRoleCredentialsRequest &request)

```

Parâmetros

[FGameLiftGetFleetRoleCredentialsRequest](#)

Valor de retorno

Informa um objeto [the section called “FGameLiftGetFleetRoleCredentialsOutcome”](#).

Exemplo

```

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_T0_TCHAR(outres.GetAssumedUserRoleArn());
result.m_assumedRoleId = UTF8_T0_TCHAR(outres.GetAssumedRoleId());
result.m_accessKeyId = UTF8_T0_TCHAR(outres.GetAccessKeyId());
result.m_secretAccessKey = UTF8_T0_TCHAR(outres.GetSecretAccessKey());
result.m_sessionToken = UTF8_T0_TCHAR(outres.GetSessionToken());
result.m_expiration = FDateTime::FromUnixTimestamp(outres.GetExpiration());
return FGameLiftGetFleetRoleCredentialsOutcome(result);
}
else {
    return FGameLiftGetFleetRoleCredentialsOutcome(FGameLiftError(outcome.GetError()));
}
#else
return
FGameLiftGetFleetRoleCredentialsOutcome(FGameLiftGetFleetRoleCredentialsResult());
#endif
}
```

SDK do servidor para Amazon GameLift Servers versão 4 e anteriores

Esta referência documenta o SDK do servidor para Amazon GameLift Servers, versão 4.x e anteriores. O SDK do servidor fornece as principais funcionalidades que seus servidores de jogos usam para se comunicar com o Amazon GameLift Servers serviço. Por exemplo, o servidor de jogos recebe solicitações do serviço para iniciar e interromper sessões de jogo e fornece atualizações regulares sobre o status das sessões de jogo ao serviço. Integre os servidores de jogos ao SDK do servidor antes de implantá-los para hospedagem.

Use esta referência de SDK de servidor para integrar seus servidores de jogos multijogador personalizados para hospedagem com Amazon GameLift Servers. Para obter orientação sobre o processo de integração, consulte [Adicione Amazon GameLift Servers ao seu servidor de jogos](#).

A versão principal mais recente do SDK do servidor para Amazon GameLift Servers é 5.x. Os seguintes recursos de hospedagem exigem atualizações para a versão 5.x:

- Amazon GameLift Servers Anywhere
- Amazon GameLift Servers contêineres gerenciados
- Amazon GameLift Servers plugin para Unreal Engine e Unity

Tópicos

- [SDK do servidor C++ para Amazon GameLift Servers 4.x -- Ações](#)
- [SDK do servidor C# para Amazon GameLift Servers 4.x -- Ações](#)
- [SDK do servidor \(Unreal\) para Amazon GameLift Servers -- Ações](#)

SDK do servidor C++ para Amazon GameLift Servers 4.x -- Ações

Use a referência do SDK do servidor para integrar seu jogo multijogador para hospedagem com Amazon GameLift Servers. Para obter orientação sobre o processo de integração, consulte [Adicione Amazon GameLift Servers ao seu servidor de jogos](#).

Note

Essa referência é para uma versão anterior do SDK do servidor para Amazon GameLift Servers. Para obter a versão mais recente, consulte [SDK 5.x do servidor C++ para Amazon GameLift Servers -- Ações](#).

SDK do servidor C++ para Amazon GameLift Servers 4.x -- Tipos de dados

Use a referência do SDK do servidor para integrar seu jogo multijogador para hospedagem com Amazon GameLift Servers. Para obter orientação sobre o processo de integração, consulte [Adicione Amazon GameLift Servers ao seu servidor de jogos](#).

Note

Essa referência é para uma versão anterior do SDK do servidor para Amazon GameLift Servers. Para obter a versão mais recente, consulte [SDK 5.x do servidor C++ para Amazon GameLift Servers -- Tipos de dados](#).

Esta API é definida em `GameLiftServerAPI.h`, `LogParameters.h` e `ProcessParameters.h`.

[SDK do servidor C++ para Amazon GameLift Servers 4.x -- Ações](#)

`DescribePlayerSessionsRequest`

Esse tipo de dados é usado para especificar quais sessões de jogador recuperar. Você pode usá-lo da seguinte forma:

- Forneça um `PlayerSessionId` para solicitar uma sessão de jogador específica.
- Forneça um `GameSessionId` para solicitar todas as sessões de jogadores na sessão de jogo especificada.
- Forneça um `PlayerId` para solicitar todas as sessões de jogadores para o jogador especificado.

Para grandes coleções de sessões de jogador, use os parâmetros de paginação para recuperar resultados em blocos sequenciais.

Conteúdo

GameSessionId

Identificador da sessão de jogo exclusivo. Use esse parâmetro para solicitar todas as sessões de jogador da sessão de jogo especificada. O formato do ID da sessão de jogo é o seguinte: `arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string>`. O valor de `<ID string>` é uma string de ID personalizada ou (se houver uma especificada quando a sessão de jogo foi criada) uma string gerada.

Tipo: String

Obrigatório: Não

Limite

O número máximo de resultados a serem retornados. Use esse parâmetro com `NextToken` para obter resultados como um conjunto de páginas sequenciais. Se um ID de sessão de jogador for especificado, esse parâmetro será ignorado.

Tipo: número inteiro

Obrigatório: não

NextToken

Token indicando o início da próxima página sequencial de resultados. Use o token retornado com uma chamada anterior para essa ação. Para especificar o início do conjunto de resultados, não especifique um valor. Se um ID de sessão de jogador for especificado, esse parâmetro será ignorado.

Tipo: String

Obrigatório: Não

PlayerId

Identificador exclusivo para um jogador. IDs Os jogadores são definidos pelo desenvolvedor. Consulte [Gerar jogador IDs](#).

Tipo: String

Obrigatório: Não

PlayerSessionId

Identificador exclusivo para uma sessão de jogador.

Tipo: String

Obrigatório: Não

PlayerSessionStatusFilter

Status da sessão de jogador para filtrar resultados. Entre os status da sessão de jogador possíveis estão os seguintes:

- RESERVED – A solicitação de sessão de jogador foi recebida, mas o jogador ainda não se conectou ao processo de servidor e/ou foi validado.
- ACTIVE – O jogador foi validado pelo processo de servidor e está conectado no momento.
- COMPLETED – A conexão do jogador foi interrompida.
- TIMEDOUT – Uma solicitação de sessão de jogador foi recebida, mas o jogador não se conectou e/ou não foi validado dentro do tempo limite (60 segundos).

Tipo: String

Obrigatório: Não

LogParameters

Esse tipo de dados é usado para identificar quais arquivos você deseja gerar durante uma sessão de jogo. Amazon GameLift Servers para carregar e armazenar quando a sessão do jogo terminar. Essas informações são comunicadas ao Amazon GameLift Servers serviço em uma [ProcessReady\(\)](#) chamada.

Conteúdo

logPaths

Caminhos de diretório para os arquivos de log do servidor de jogos que você deseja Amazon GameLift Servers para armazenar para acesso futuro. Esses arquivos são gerados durante cada sessão do jogo. Os nomes e caminhos de arquivos são definidos no seu servidor de jogos e armazenados no diretório raiz da compilação do jogo. Os caminhos de log devem ser caminhos absolutos. Por exemplo, se a compilação do jogo armazenasse logs de sessão de jogo em um caminho como `MyGame\sessionlogs\`, o caminho de log seria `c:\game\MyGame\sessionLogs` (em uma instância do Windows) ou `/local/game/MyGame/sessionLogs` (em uma instância do Linux).

Tipo: `std::vector<std::string>`

Obrigatório: não

ProcessParameters

Esse tipo de dados contém o conjunto de parâmetros enviados para o Amazon GameLift Servers serviço em uma [ProcessReady\(\)](#) chamada.

Conteúdo

porta

Número da porta em que o processo de servidor escuta para novas conexões de jogador. O valor deve estar dentro do intervalo de portas configurado para qualquer frota que implanta essa compilação do servidor de jogo. Esse número de porta está incluído nos objetos das sessões de jogo e jogador, que as sessões de jogo usam ao se conectar a um processo de servidor.

Tipo: inteiro

Obrigatório: sim

logParameters

Objeto com uma lista de caminhos de diretório para arquivos de log da sessão de jogo.

Tipo: `Aws::GameLift::Servidor::` [LogParameters](#)

Obrigatório: não

onStartGameSessão

Nome da função de retorno de chamada que o Amazon GameLift Servers o serviço é invocado para ativar uma nova sessão de jogo. Amazon GameLift Servers chama essa função em resposta à solicitação do cliente [CreateGameSession](#). A função de retorno de chamada passa um [GameSession](#) objeto (definido no Amazon GameLift Servers Referência da API de serviço).

Tipo: `const std::function<void(Aws::GameLift::Model::GameSession)>`
`onStartGameSession`

Obrigatório: Sim

onProcessTerminate

Nome da função de retorno de chamada que o Amazon GameLift Servers o serviço é invocado para forçar o encerramento do processo do servidor. Depois de chamar essa função, Amazon GameLift Servers espera cinco minutos até que o processo do servidor seja encerrado e responda com uma [ProcessEnding\(\)](#) chamada. Se nenhuma resposta for recebida, ele desativará o processo do servidor.

Tipo: `std::function<void()>` `onProcessTerminate`

Obrigatório: não

onHealthCheck

Nome da função de retorno de chamada que o Amazon GameLift Servers o serviço é chamado para solicitar um relatório de status de saúde do processo do servidor. Amazon GameLift Servers chama essa função a cada 60 segundos. Depois de chamar esta função Amazon GameLift Servers espera 60 segundos por uma resposta e, se nenhuma for recebida, registra o processo do servidor como não íntegro.

Tipo: `std::function<bool()>` `onHealthCheck`

Obrigatório: não

onUpdateGameSessão

Nome da função de retorno de chamada que o Amazon GameLift Servers o serviço é invocado para passar um objeto de sessão de jogo atualizado para o processo do servidor. Amazon GameLift Servers chama essa função quando uma solicitação de [preenchimento de correspondência](#) é processada para fornecer dados atualizados do matchmaker. Ele transmite um

objeto [GameSession](#), uma atualização de status (updateReason) e o ID do tíquete de alocação de correspondência.

Tipo: `std::function<void(Aws::GameLift::Server::Model::UpdateGameSession)>`
`onUpdateGameSession`

Obrigatório: não

StartMatchBackfillRequest

Esse tipo de dados é usado para enviar uma solicitação de alocação de marcação de jogos. As informações são comunicadas ao Amazon GameLift Servers serviço em uma [StartMatchBackfill\(\)](#) chamada.

Conteúdo

GameSessionArn

Identificador da sessão de jogo exclusivo. A ação de API [GetGameSessionId\(\)](#) retorna o identificador no formato de ARN.

Tipo: sequência

Obrigatório: Sim

MatchmakingConfigurationArn

O identificador exclusivo, no formato de ARN, para o marcador de jogos a ser usado para essa solicitação. Para encontrar o marcador de jogos usado para criar a sessão de jogo original, procure no objeto de sessão do jogo, na propriedade de dados do marcador de jogos. Saiba mais sobre os dados do marcador de jogos no [Trabalhar com dados do marcador de jogos](#).

Tipo: sequência

Obrigatório: Sim

Jogadores

Um conjunto de dados que representa todos os jogadores que estão na sessão do jogo no momento. O marcador de jogos usa essas informações para pesquisar novos jogadores que sejam correspondências ideais para os jogadores atuais. Veja o Amazon GameLift Servers Guia de referência da API para uma descrição do formato do objeto Player. Para encontrar

os atributos do jogador e as atribuições da equipe, procure no objeto da sessão do jogo, na propriedade de dados do matchmaker. IDs Se o marcador de jogos usa latência, obtenha a latência atualizada para a região atual e a inclua nos dados de cada jogador.

Tipo: `std::vector<player>`

Obrigatório: sim

TicketId

O identificador exclusivo de um tíquete de solicitação de marcação de jogo ou de alocação de correspondência. Se nenhum valor for fornecido aqui, Amazon GameLift Servers gerará um na forma de um UUID. Use esse identificador para acompanhar o status do tíquete de alocação de correspondência ou cancelar a solicitação, se necessário.

Tipo: String

Obrigatório: Não

StopMatchBackfillRequest

Esse tipo de dados é usado para cancelar uma solicitação de alocação de marcação de jogo. As informações são comunicadas ao Amazon GameLift Servers serviço em uma [StopMatchBackfill\(\)](#) chamada.

Conteúdo

GameSessionArn

O identificador de sessão de jogo exclusivo associado à solicitação que está sendo cancelada.

Tipo: sequência

Obrigatório: Sim

MatchmakingConfigurationArn

O identificador exclusivo do marcador de jogos para o qual essa solicitação foi enviada.

Tipo: sequência

Obrigatório: Sim

TicketId

O identificador exclusivo do tíquete de solicitação de alocação a ser cancelado.

Tipo: sequência

Obrigatório: Sim

[SDK do servidor C++ para Amazon GameLift Servers 4.x -- Tipos de dados](#)

Tópicos

- [AcceptPlayerSession\(\)](#)
- [ActivateGameSession\(\)](#)
- [DescribePlayerSessions\(\)](#)
- [GetGameSessionId\(\)](#)
- [GetInstanceCertificate\(\)](#)
- [GetSdkVersion\(\)](#)
- [GetTerminationTime\(\)](#)
- [InitSDK\(\)](#)
- [ProcessEnding\(\)](#)
- [ProcessReady\(\)](#)
- [ProcessReadyAsync\(\)](#)
- [RemovePlayerSession\(\)](#)
- [StartMatchBackfill\(\)](#)
- [StopMatchBackfill\(\)](#)
- [TerminateGameSession\(\)](#)
- [UpdatePlayerSessionCreationPolicy\(\)](#)
- [Destroy\(\)](#)

AcceptPlayerSession()

Notifica o Amazon GameLift Servers serviço de que um jogador com o ID de sessão de jogador especificado se conectou ao processo do servidor e precisa de validação. Amazon GameLift Servers verifica se o ID da sessão do jogador é válido, ou seja, se o ID do jogador reservou um

espaço para o jogador na sessão do jogo. Depois de validado, o Amazon GameLift Servers altera o status do slot do jogador de RESERVED para ACTIVE.

Sintaxe

```
GenericOutcome AcceptPlayerSession(const std::string& playerSessionId);
```

Parâmetros

playerSessionId

ID exclusiva emitida pelo Amazon GameLift Servers serviço em resposta a uma chamada para a ação [CreatePlayerSession](#) da Amazon GameLift Servers API do AWS SDK. O cliente do jogo faz referência a esse ID ao se conectar ao processo do servidor.

Tipo: std::string

Obrigatório: Sim

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

Este exemplo ilustra uma função para lidar com uma solicitação de conexão, incluindo a validação e rejeição de uma sessão de jogador inválida. 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 o serviço Amazon GameLift Servers de que o processo do servidor iniciou uma sessão de jogo e está pronto para receber as conexões dos jogadores. Essa ação deve ser chamada como parte da função de retorno de chamada `onStartGameSession()`, depois que toda a inicialização da sessão tiver sido concluída.

Sintaxe

```
GenericOutcome ActivateGameSession();
```

Parâmetros

Essa ação não tem um parâmetro.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

Este exemplo mostra `ActivateGameSession()` sendo chamado como parte da função de retorno de chamada `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 dados da sessão do jogador, inclusive configurações, metadados da sessão e dados do jogador. Use essa ação para obter informações de uma única sessão de jogador, para todas as sessões de jogador em uma sessão de jogo, ou para todas as sessões de jogador associadas a um único ID de jogador.

Sintaxe

```
DescribePlayerSessionsOutcome DescribePlayerSessions (
```

```
const Aws::GameLift::Server::Model::DescribePlayerSessionsRequest
&describePlayerSessionsRequest);
```

Parâmetros

describePlayerSessionsSolicitação

Um objeto [DescribePlayerSessionsRequest](#) que descreve quais sessões de jogador recuperar.

Obrigatório: Sim

Valor de retorno

Se bem-sucedido, retorna um objeto `DescribePlayerSessionsOutcome` que contém um conjunto de objetos de sessão do jogador que atendem aos parâmetros de solicitação. Os objetos de sessão do player têm uma estrutura idêntica ao tipo de [PlayerSession](#) dados da Amazon GameLift Servers API do AWS SDK.

Exemplo

Este exemplo ilustra uma solicitação para todas as sessões de jogador conectadas ativamente a uma sessão de jogo especificada. Omitindo `NextToken` e definindo o valor `Limit` como 10, o Amazon GameLift Servers retorna os 10 primeiros registros de sessões do jogador correspondentes à solicitação.

```
// 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 um identificador exclusivo para a sessão do jogo hospedada no momento pelo processo do servidor, caso o processo do servidor esteja ativo. O identificador é retornado no formato de ARN: `arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string>`.

Para processos inativos que ainda não foram ativados com uma sessão de jogo, a chamada retorna `Success=True` e `GameSessionId=""` (uma string vazia).

Sintaxe

```
AwsStringOutcome GetGameSessionId();
```

Parâmetros

Essa ação não tem um parâmetro.

Valor de retorno

Se bem-sucedido, retornará o ID da sessão de jogo como um objeto `AwsStringOutcome`. Se não for bem-sucedido, retornará uma mensagem de erro.

Exemplo

```
Aws::GameLift::AwsStringOutcome sessionIdOutcome =  
    Aws::GameLift::Server::GetGameSessionId();
```

GetInstanceCertificate()

Recupera a localização do arquivo de um certificado TLS codificado por pem que está associado à frota e suas instâncias. AWS Certificate Manager gera esse certificado quando você cria uma nova frota com a configuração do certificado definida como `GENERATED`. Use esse certificado para estabelecer uma conexão segura com um cliente de jogo e para criptografar a comunicação entre cliente e servidor.

Sintaxe

```
GetInstanceCertificateOutcome GetInstanceCertificate();
```

Parâmetros

Essa ação não tem um parâmetro.

Valor de retorno

Se for bem-sucedido, o retorno será um objeto `GetInstanceCertificateOutcome` que contém a localização do arquivo de certificado TLS e da cadeia de certificados da frota, que estão

armazenados na instância. Um arquivo de certificado raiz, extraído da cadeia de certificados, também é armazenado na instância. Se não for bem-sucedido, retornará uma mensagem de erro.

Para obter mais informações sobre o certificado e os dados da cadeia de certificados, consulte [Elementos de GetCertificate resposta](#) na Referência da AWS Certificate Manager API.

Exemplo

```
Aws::GameLift::GetInstanceCertificateOutcome certificateOutcome =  
    Aws::GameLift::Server::GetInstanceCertificate();
```

GetSdkVersion()

Retorna o número da versão atual do SDK em uso.

Sintaxe

```
AwsStringOutcome GetSdkVersion();
```

Parâmetros

Essa ação não tem um parâmetro.

Valor de retorno

Se bem-sucedido, retornará a versão do SDK atual como um objeto `AwsStringOutcome`. A string retornada inclui apenas o número da versão (exemplo, "3.1.5"). Se não for bem-sucedido, retornará uma mensagem de erro.

Exemplo

```
Aws::GameLift::AwsStringOutcome SdkVersionOutcome =  
    Aws::GameLift::Server::GetSdkVersion();
```

GetTerminationTime()

Retorna a hora em que um processo do servidor está programado para ser desligado, se essa informação estiver disponível. Um processo do servidor executa essa ação depois de receber um `onProcessTerminate()` retorno de chamada do Amazon GameLift Servers serviço. Amazon

GameLift Servers [pode ligar `onProcessTerminate\(\)` pelos seguintes motivos: \(1\) quando o processo do servidor relatou problemas de saúde ou não respondeu](#) Amazon GameLift Servers, [\(2\) ao encerrar a instância durante um evento de redução de escala ou \(3\) quando uma instância está sendo encerrada devido a uma interrupção do Spot.](#)

Se o processo tiver recebido um retorno de chamada `onProcessTerminate()`, o valor retornado será o tempo estimado de encerramento. Se o processo não tiver recebido um retorno de chamada `onProcessTerminate()`, uma mensagem de erro será retornada. Saiba mais sobre [como desligar um processo do servidor](#).

Sintaxe

```
AwsLongOutcome GetTerminationTime();
```

Parâmetros

Essa ação não tem um parâmetro.

Valor de retorno

Se for bem-sucedido, retornará o horário de término como um objeto `AwsLongOutcome`. O valor é o tempo de término, expresso em tiques decorridos desde 0001 00:00:00. Por exemplo, o valor da data e hora 2020-09-13 12:26:40 -000Z é igual aos tiques 637355968000000000. Se nenhum horário de rescisão estiver disponível, o retornará uma mensagem de erro.

Exemplo

```
Aws::GameLift::AwsLongOutcome TermTimeOutcome =  
    Aws::GameLift::Server::GetTerminationTime();
```

InitSDK()

Inicializa o Amazon GameLift Servers SDK. Esse método deve ser chamado na inicialização antes de qualquer outra inicialização relacionada ao Amazon GameLift Servers.

Sintaxe

```
InitSDKOutcome InitSDK();
```

Parâmetros

Essa ação não tem um parâmetro.

Valor de retorno

Se for bem-sucedido, retornará um `InitSdkOutcome` objeto indicando que o processo do servidor está pronto para ser chamado [ProcessReady\(\)](#).

Exemplo

```
Aws::GameLift::Server::InitSDKOutcome initOutcome =  
    Aws::GameLift::Server::InitSDK();
```

ProcessEnding()

Notifica o serviço Amazon GameLift Servers de que o processo do servidor está sendo desligado. Esse método deverá ser chamado depois de todas as outras tarefas de limpeza, inclusive desligar todas as sessões de jogos ativas. Este método deve sair com um código de saída zero; um código de saída diferente de zero resulta em uma mensagem de evento indicando que o processo não foi encerrado corretamente.

Depois que o método sair com um código de 0, você poderá encerrar o processo com um código de saída bem-sucedido. Você também poderá sair do processo com um código de erro. Se você sair com um código de erro, o evento da frota indicará que o processo foi encerrado de forma anormal (`SERVER_PROCESS_TERMINATED_UNHEALTHY`).

Sintaxe

```
GenericOutcome ProcessEnding();
```

Parâmetros

Essa ação não tem um parâmetro.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

```
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 o serviço Amazon GameLift Servers de que o processo do servidor está pronto para hospedar sessões do jogo. Chame esse método depois de invocar [InitSDK\(\)](#) e concluir todas as tarefas de configuração necessárias antes que o processo do servidor possa hospedar uma sessão do jogo. Esse método deve ser chamado somente uma vez por processo.

Essa chamada é síncrona. Para fazer uma chamada assíncrona, use [ProcessReadyAsync\(\)](#). Consulte [Inicializar o processo do servidor](#) para obter mais detalhes.

Sintaxe

```
GenericOutcome ProcessReady(
    const Aws::GameLift::Server::ProcessParameters &processParameters);
```

Parâmetros

processParameters

Um objeto [ProcessParameters](#) que fornece as seguintes informações sobre o processo do servidor:

- Nomes de métodos de retorno de chamada, implementados no código do servidor de jogos, que o serviço Amazon GameLift Servers invoca para se comunicar com o processo de servidor.
- Número da porta em que o processo de servidor está escutando.
- Caminho de qualquer arquivo específico da sessão do jogo que você deseja que o Amazon GameLift Servers capture e armazene.

Obrigatório: Sim

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

Este exemplo ilustra as implementações das funções de chamada e retorno de chamada [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 o serviço Amazon GameLift Servers de que o processo do servidor está pronto para hospedar sessões do jogo. Este método deve ser chamado assim que o processo do servidor estiver pronto para hospedar uma sessão de jogo. Os parâmetros especificam os nomes das funções de retorno do Amazon GameLift Servers para chamar determinadas circunstâncias. O código do servidor de jogos deve implementar essas funções.

Essa chamada é assíncrona. Para fazer uma chamada síncrona, use [ProcessReady\(\)](#). Consulte [Inicializar o processo do servidor](#) para obter mais detalhes.

Sintaxe

```
GenericOutcomeCallable ProcessReadyAsync(  
    const Aws::GameLift::Server::ProcessParameters &processParameters);
```

Parâmetros

processParameters

Um objeto [ProcessParameters](#) que fornece as seguintes informações sobre o processo do servidor:

- Nomes de métodos de retorno de chamada, implementados no código do servidor de jogos, que o serviço Amazon GameLift Servers invoca para se comunicar com o processo de servidor.
- Número da porta em que o processo de servidor está escutando.
- Caminho de qualquer arquivo específico da sessão do jogo que você deseja que o Amazon GameLift Servers capture e armazene.

Obrigatório: Sim

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

```
// 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 o serviço Amazon GameLift Servers de que um jogador com o ID da sessão do jogador especificado se desconectou do processo do servidor. Em resposta, o Amazon GameLift Servers altera o slot do jogador para um disponível, o que permite ser atribuído a um novo jogador.

Sintaxe

```
GenericOutcome RemovePlayerSession(  
    const std::string& playerId);
```

Parâmetros

playerSessionId

ID exclusiva emitida pelo Amazon GameLift Servers serviço em resposta a uma chamada para a ação [CreatePlayerSession](#) da Amazon GameLift Servers API do AWS SDK. O cliente do jogo faz referência a esse ID ao se conectar ao processo do servidor.

Tipo: std::string

Obrigatório: Sim

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

```
Aws::GameLift::GenericOutcome disconnectOutcome =  
    Aws::GameLift::Server::RemovePlayerSession(playerSessionId);
```

StartMatchBackfill()

Envia uma solicitação para encontrar novos jogadores para os slots abertos em uma sessão de jogo criada com o FlexMatch. Veja também a ação do AWS SDK [StartMatchBackfill\(\)](#). Com essa ação, as solicitações de alocação de correspondência podem ser iniciadas por um processo do servidor de jogos que esteja hospedando a sessão do jogo. Saiba mais sobre o [recurso FlexMatch de preenchimento](#).

Esta ação é assíncrona. Se novos jogadores forem combinados com sucesso, o Amazon GameLift Servers serviço fornece dados atualizados do matchmaker invocando a função de retorno de chamada. `OnUpdateGameSession()`

Um processo de servidor pode ter apenas uma solicitação de alocação de correspondência ativa por vez. Para enviar uma nova solicitação, primeiro chame [StopMatchBackfill\(\)](#) para cancelar a solicitação original.

Sintaxe

```
StartMatchBackfillOutcome StartMatchBackfill (
    const Aws::GameLift::Server::Model::StartMatchBackfillRequest
    &startBackfillRequest);
```

Parâmetros

StartMatchBackfillRequest

Um objeto [StartMatchBackfillRequest](#) que fornece as seguintes informações:

- ID do tíquete a ser atribuído à solicitação de alocação. Essa informação é opcional. Caso nenhum ID seja fornecido, o Amazon GameLift Servers gerará um automaticamente.
- O marcador de jogos para o qual a solicitação é enviada. O ARN completo da configuração é necessário. Esse valor pode ser obtido dos dados do marcador da sessão do jogo.
- ID da sessão de jogo que está sendo alocada.
- Dados disponíveis de marcação para os jogadores atuais da sessão do jogo.

Obrigatório: Sim

Valor de retorno

Retorna um `StartMatchBackfillOutcome` objeto com o tíquete de preenchimento da partida ou uma falha com uma mensagem de erro. O status do ticket pode ser monitorado usando a ação do AWS SDK [DescribeMatchmaking\(\)](#).

Exemplo

```
// 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()

Cancela uma solicitação de alocação de correspondência ativa criada com [StartMatchBackfill\(\)](#). Veja também a ação do AWS SDK [StopMatchmaking\(\)](#). Saiba mais sobre o [recurso FlexMatch de preenchimento](#).

Sintaxe

```

GenericOutcome StopMatchBackfill (
    const Aws::GameLift::Server::Model::StopMatchBackfillRequest &stopBackfillRequest);

```

Parâmetros

StopMatchBackfillRequest

Um objeto [StopMatchBackfillRequest](#) que identifica o tíquete de marcação de jogos para cancelar:

- ID do tíquete atribuído à solicitação de alocação sendo cancelada
- o marcador de jogo que recebeu a solicitação de alocação

- a sessão do jogo associada à solicitação de alocação

Obrigatório: Sim

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

```
// 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()

Esse método foi descontinuado com a versão 4.0.1. Em vez disso, o processo do servidor deve ser chamado [ProcessEnding\(\)](#) após o término de uma sessão de jogo.

Notifica o Amazon GameLift Servers serviço de que o processo do servidor encerrou a sessão atual do jogo. Essa ação é chamada quando o processo do servidor permanece ativo e pronto para hospedar uma nova sessão de jogo. Ele deve ser chamado somente após a conclusão do procedimento de encerramento da sessão de jogo, pois indica Amazon GameLift Servers que o processo do servidor está imediatamente disponível para hospedar uma nova sessão de jogo.

Essa ação não será chamada se o processo do servidor for encerrado após o término da sessão do jogo. Em vez disso, chame [ProcessEnding\(\)](#) para sinalizar que tanto a sessão do jogo quanto o processo do servidor estão terminando.

Sintaxe

```
GenericOutcome TerminateGameSession();
```

Parâmetros

Essa ação não tem um parâmetro.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

UpdatePlayerSessionCreationPolicy()

Atualiza a capacidade da sessão do jogo atual para aceitar novas sessões de jogador. Atualiza a capacidade da sessão do jogo atual para aceitar novas sessões de jogador. Veja também a ação do AWS SDK [UpdateGameSession\(\)](#).

Sintaxe

```
GenericOutcome UpdatePlayerSessionCreationPolicy(  
    Aws::GameLift::Model::PlayerSessionCreationPolicy newPlayerSessionPolicy);
```

Parâmetros

newPlayerSessionPolítica

Valor de string que indica se a sessão do jogo aceita novos jogadores.

Tipo: `Aws::GameLift::Model::PlayerSessionCreationPolicy` enum. Os valores válidos são:

- `ACCEPT_ALL` – Aceite todas as novas sessões de jogador.
- `DENY_ALL` – Recuse todas as novas sessões de jogador.

Obrigatório: Sim

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

Este exemplo define a política de ingresso da sessão do jogo atual para aceitar todos os jogadores.

```
Aws::GameLift::GenericOutcome outcome =  
    Aws::GameLift::Server::UpdatePlayerSessionCreationPolicy(Aws::GameLift::Model::PlayerSessionCr
```


Destroy()

Limpa a memória alocada por `initSDK ()` durante a inicialização do servidor do jogo. Use esse método depois de finalizar um processo no servidor do jogo para evitar o desperdício de memória do servidor.

Sintaxe

```
GenericOutcome Aws::GameLift::Server::Destroy();
```

Parâmetros

Não há parâmetros.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

Este exemplo limpa a memória alocada pelo `initSDK` após o término de um processo no servidor de jogos.

```
if (Aws::GameLift::Server::ProcessEnding().IsSuccess()) {  
    Aws::GameLift::Server::Destroy();  
    exit(0);  
}
```

SDK do servidor C# para Amazon GameLift Servers 4.x -- Ações

Use a referência do SDK do servidor para integrar seu jogo multijogador para hospedagem com Amazon GameLift Servers. Para obter orientação sobre o processo de integração, consulte [Adicione Amazon GameLift Servers ao seu servidor de jogos](#).

Note

Essa referência é para uma versão anterior do SDK do servidor para Amazon GameLift Servers. Para obter a versão mais recente, consulte [SDK 5.x do servidor C# para Amazon GameLift Servers -- Ações](#).

SDK do servidor C# para Amazon GameLift Servers 4.x -- Tipos de dados

Use a referência do SDK do servidor para integrar seu jogo multijogador para hospedagem com Amazon GameLift Servers. Para obter orientação sobre o processo de integração, consulte [Adicione Amazon GameLift Servers ao seu servidor de jogos](#).

Note

Essa referência é para uma versão anterior do SDK do servidor para Amazon GameLift Servers. Para obter a versão mais recente, consulte [SDK 5.x do servidor C# para Amazon GameLift Servers -- Tipos de dados](#).

[SDK do servidor C# para Amazon GameLift Servers 4.x -- Ações](#)

Tópicos

- [LogParameters](#)
- [DescribePlayerSessionsRequest](#)
- [ProcessParameters](#)
- [StopMatchBackfillRequest](#)

LogParameters

Esse tipo de dados é usado para identificar quais arquivos você deseja gerar durante uma sessão de jogo Amazon GameLift Servers para carregar e armazenar quando a sessão do jogo terminar. Essas informações são comunicadas ao Amazon GameLift Servers serviço em uma [ProcessReady\(\)](#) chamada.

Conteúdo

logPaths

Lista de caminhos de diretório para os arquivos de log do servidor de jogos que você deseja Amazon GameLift Servers para armazenar para acesso futuro. Esses arquivos são gerados por um processo de servidor durante cada sessão de jogo; caminhos de arquivo e nomes são definidos no servidor de jogo e armazenados no diretório de compilação do jogo raiz. Os caminhos de registro devem ser caminhos absolutos. Por exemplo, se a compilação do jogo

armazenasse logs de sessão de jogo em um caminho como `MyGame\sessionlogs\`, o caminho de log seria `c:\game\MyGame\sessionLogs` (em uma instância do Windows) ou `/local/game/MyGame/sessionLogs` (em uma instância do Linux).

Tipo: `List<String>`

Obrigatório: não

DescribePlayerSessionsRequest

Esse tipo de dados é usado para especificar quais sessões de jogador recuperar. Ele pode ser usado de várias maneiras: (1) forneça `PlayerSessionId` a para solicitar uma sessão de um jogador específico; (2) forneça `GameSessionId` a para solicitar todas as sessões de jogadores na sessão de jogo especificada; ou (3) forneça `PlayerId` a para solicitar todas as sessões de jogadores para o jogador especificado. Para coleções maiores de sessões de jogador, use os parâmetros de paginação para recuperar resultados como páginas sequenciais.

Conteúdo

GameSessionId

Identificador da sessão de jogo exclusivo. Use esse parâmetro para solicitar todas as sessões de jogador da sessão de jogo especificada. O formato do ID da sessão de jogo é o seguinte: `arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string>`. O valor de `<ID string>` é uma string de ID personalizada (se uma tiver sido especificada quando a sessão do jogo foi criada) ou uma string gerada.

Tipo: `String`

Obrigatório: Não

Limite

O número máximo de resultados a serem retornados. Use esse parâmetro com `NextToken` para obter resultados como um conjunto de páginas sequenciais. Se um ID de sessão de jogador for especificado, esse parâmetro será ignorado.

Tipo: número inteiro

Obrigatório: não

NextToken

Token indicando o início da próxima página sequencial de resultados. Use o token retornado com uma chamada anterior para essa ação. Para especificar o início do conjunto de resultados, não especifique um valor. Se um ID de sessão de jogador for especificado, esse parâmetro será ignorado.

Tipo: String

Obrigatório: Não

PlayerId

Identificador exclusivo para um jogador. IDs Os jogadores são definidos pelo desenvolvedor. Consulte [Gerar jogador IDs](#).

Tipo: String

Obrigatório: Não

PlayerSessionId

Identificador exclusivo para uma sessão de jogador.

Tipo: String

Obrigatório: Não

PlayerSessionStatusFilter

Status da sessão de jogador para filtrar resultados. Entre os status da sessão de jogador possíveis estão os seguintes:

- RESERVED – A solicitação de sessão de jogador foi recebida, mas o jogador ainda não se conectou ao processo de servidor e/ou foi validado.
- ACTIVE – O jogador foi validado pelo processo de servidor e está conectado no momento.
- COMPLETED – A conexão do jogador foi interrompida.
- TIMEDOUT – Uma solicitação de sessão de jogador foi recebida, mas o jogador não se conectou e/ou não foi validado dentro do tempo limite (60 segundos).

Tipo: String

Obrigatório: Não

ProcessParameters

Esse tipo de dados contém o conjunto de parâmetros enviados ao Amazon GameLift Servers serviço em uma [ProcessReady\(\)](#) chamada.

Conteúdo

porta

Número da porta em que o processo de servidor escutará novas conexões de jogador. O valor deve estar dentro do intervalo de portas configurado para qualquer frota que implanta essa compilação do servidor de jogo. Esse número de porta está incluído nos objetos das sessões de jogo e jogador, que as sessões de jogo usam ao se conectar a um processo de servidor.

Tipo: inteiro

Obrigatório: sim

logParameters

Objeto com uma lista de caminhos de diretório para arquivos de log da sessão de jogo.

Tipo: `Aws::GameLift::Server::LogParameters`

Obrigatório: Sim

onStartGameSessão

Nome da função de retorno de chamada que o Amazon GameLift Servers o serviço é invocado para ativar uma nova sessão de jogo. Amazon GameLift Servers chama essa função em resposta à solicitação do cliente [CreateGameSession](#). A função de retorno de chamada usa um [GameSession](#) objeto (definido no Amazon GameLift Servers Referência da API de serviço).

Tipo: `void OnStartGameSessionDelegate(GameSession gameSession)`

Obrigatório: Sim

onProcessTerminate

Nome da função de retorno de chamada que o Amazon GameLift Servers o serviço é invocado para forçar o encerramento do processo do servidor. Depois de chamar essa função, Amazon GameLift Servers espera cinco minutos até que o processo do servidor seja encerrado e responda com uma [ProcessEnding\(\)](#) chamada antes de encerrar o processo do servidor.

Tipo: `void OnProcessTerminateDelegate()`

Obrigatório: Sim

onHealthCheck

Nome da função de retorno de chamada que o Amazon GameLift Servers o serviço é chamado para solicitar um relatório de status de saúde do processo do servidor. Amazon GameLift Servers chama essa função a cada 60 segundos. Depois de chamar esta função Amazon GameLift Servers espera 60 segundos por uma resposta e, se nenhuma for recebida, registra o processo do servidor como não íntegro.

Tipo: `bool OnHealthCheckDelegate()`

Obrigatório: Sim

onUpdateGameSessão

Nome da função de retorno de chamada que o Amazon GameLift Servers o serviço é invocado para passar um objeto de sessão de jogo atualizado para o processo do servidor. Amazon GameLift Servers chama essa função quando uma solicitação de [preenchimento de correspondência](#) é processada para fornecer dados atualizados do matchmaker. Ele transmite um objeto [GameSession](#), uma atualização de status (`updateReason`) e o ID do tíquete de alocação de correspondência.

Tipo: `void OnUpdateGameSessionDelegate (UpdateGameSession
updateGameSession)`

Obrigatório: não

StartMatchBackfillRequest

Esse tipo de dados é usado para enviar uma solicitação de alocação de marcação de jogos. As informações são comunicadas ao Amazon GameLift Servers serviço em uma [StartMatchBackfill\(\)](#) chamada.

Conteúdo

GameSessionArn

Identificador da sessão de jogo exclusivo. O método [GetGameSessionId\(\)](#) do SDK retorna o identificador no formato de ARN.

Tipo: sequência

Obrigatório: Sim

MatchmakingConfigurationArn

O identificador exclusivo, no formato de ARN, para o marcador de jogos a ser usado para essa solicitação. Para encontrar o marcador de jogos usado para criar a sessão de jogo original, procure no objeto de sessão do jogo, na propriedade de dados do marcador de jogos. Saiba mais sobre os dados de marcador de jogos em [Trabalhar com dados do marcador de jogos](#).

Tipo: sequência

Obrigatório: Sim

Jogadores

Um conjunto de dados que representa todos os jogadores que estão na sessão do jogo no momento. O marcador de jogos usa essas informações para pesquisar novos jogadores que sejam correspondências ideais para os jogadores atuais. Veja o Amazon GameLift Servers Guia de referência da API para uma descrição do formato do objeto Player. Para encontrar os atributos do jogador e as atribuições da equipe, procure no objeto da sessão do jogo, na propriedade de dados do matchmaker. IDs Se o marcador de jogos usa latência, obtenha a latência atualizada para a região atual e a inclua nos dados de cada jogador.

Tipo: [Player](#)[]

Obrigatório: sim

TicketId

O identificador exclusivo de um tíquete de solicitação de marcação de jogo ou de alocação de correspondência. Se nenhum valor for fornecido aqui, Amazon GameLift Servers gerará um na forma de um UUID. Use esse identificador para acompanhar o status do tíquete de alocação de correspondência ou cancelar a solicitação, se necessário.

Tipo: String

Obrigatório: Não

StopMatchBackfillRequest

Esse tipo de dados é usado para cancelar uma solicitação de alocação de marcação de jogo. As informações são comunicadas ao Amazon GameLift Servers serviço em uma [StopMatchBackfill\(\)](#) chamada.

Conteúdo

GameSessionArn

O identificador de sessão de jogo exclusivo associado à solicitação que está sendo cancelada.

Tipo: sequência

Obrigatório: Sim

MatchmakingConfigurationArn

O identificador exclusivo do marcador de jogos para o qual essa solicitação foi enviada.

Tipo: sequência

Obrigatório: Sim

TicketId

O identificador exclusivo do tíquete de solicitação de alocação a ser cancelado.

Tipo: sequência

Obrigatório: Sim

[SDK do servidor C# para Amazon GameLift Servers 4.x -- Tipos de dados](#)

Tópicos

- [AcceptPlayerSession\(\)](#)
- [ActivateGameSession\(\)](#)
- [DescribePlayerSessions\(\)](#)
- [GetGameSessionId\(\)](#)
- [GetInstanceCertificate\(\)](#)
- [GetSdkVersion\(\)](#)
- [GetTerminationTime\(\)](#)
- [InitSDK\(\)](#)
- [ProcessEnding\(\)](#)

- [ProcessReady\(\)](#)
- [RemovePlayerSession\(\)](#)
- [StartMatchBackfill\(\)](#)
- [StopMatchBackfill\(\)](#)
- [TerminateGameSession\(\)](#)
- [UpdatePlayerSessionCreationPolicy\(\)](#)

AcceptPlayerSession()

Notifica o Amazon GameLift Servers serviço de que um jogador com o ID de sessão de jogador especificado se conectou ao processo do servidor e precisa de validação. Amazon GameLift Servers verifica se o ID da sessão do jogador é válido, ou seja, se o ID do jogador reservou um espaço para o jogador na sessão do jogo. Depois de validado, o Amazon GameLift Servers altera o status do slot do jogador de RESERVED para ACTIVE.

Sintaxe

```
GenericOutcome AcceptPlayerSession(String playerId)
```

Parâmetros

playerSessionId

ID exclusivo emitido Amazon GameLift Servers quando uma nova sessão de jogador é criada. O ID da sessão do jogador é especificado em um `PlayerSession` objeto, que é retornado em resposta a uma chamada do cliente para as ações da GameLift API [StartGameSessionPlacement](#), [CreateGameSession](#), [DescribeGameSessionPlacement](#), ou [DescribePlayerSessions](#).

Tipo: string

Obrigatório: Sim

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

Este exemplo ilustra uma função para lidar com uma solicitação de conexão, incluindo a validação e rejeição de uma sessão de jogador inválida. 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 o serviço Amazon GameLift Servers de que o processo do servidor ativou uma sessão do jogo e já está pronto para receber as conexões do jogador. Essa ação deve ser chamada como parte da função de retorno de chamada `onStartGameSession()`, depois que toda a inicialização da sessão tiver sido concluída.

Sintaxe

```
GenericOutcome ActivateGameSession()
```

Parâmetros

Essa ação não tem um parâmetro.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

Este exemplo mostra `ActivateGameSession()` sendo chamado como parte da função de delegação `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 dados da sessão do jogador, inclusive configurações, metadados da sessão e dados do jogador. Use essa ação para obter informações de uma única sessão de jogador, para todas as sessões de jogador em uma sessão de jogo, ou para todas as sessões de jogador associadas a um único ID de jogador.

Sintaxe

```
DescribePlayerSessionsOutcome DescribePlayerSessions(DescribePlayerSessionsRequest
describePlayerSessionsRequest)
```

Parâmetros

describePlayerSessionsSolicitação

Um objeto [DescribePlayerSessionsRequest](#) que descreve quais sessões de jogador recuperar.

Obrigatório: Sim

Valor de retorno

Se bem-sucedido, retorna um objeto `DescribePlayerSessionsOutcome` que contém um conjunto de objetos de sessão do jogador que atendem aos parâmetros de solicitação. Os objetos de sessão do player têm uma estrutura idêntica ao tipo de [PlayerSession](#) dados da Amazon GameLift Servers API do AWS SDK.

Exemplo

Este exemplo ilustra uma solicitação para todas as sessões de jogador conectadas ativamente a uma sessão de jogo especificada. Ao omitir `NextToken` definir o valor limite como 10, Amazon

GameLift Servers retornará os registros das primeiras 10 sessões de jogadores que correspondam à solicitação.

```
// 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 o ID da sessão do jogo hospedada no momento pelo processo do servidor, caso o processo do servidor esteja ativo.

Para processos inativos que ainda não foram ativados com uma sessão de jogo, a chamada retorna `Success=True` e `GameSessionId=""` (uma string vazia).

Sintaxe

```
AwsStringOutcome GetGameSessionId()
```

Parâmetros

Essa ação não tem um parâmetro.

Valor de retorno

Se bem-sucedido, retornará o ID da sessão de jogo como um objeto `AwsStringOutcome`. Se não for bem-sucedido, retornará uma mensagem de erro.

Exemplo

```
var getGameSessionIdOutcome = GameLiftServerAPI.GetGameSessionId();
```

GetInstanceCertificate()

Recupera a localização do arquivo de um certificado TLS codificado por pem que está associado à frota e suas instâncias. AWS Certificate Manager gera esse certificado quando você cria uma nova frota com a configuração do certificado definida como GENERATED. Use esse certificado para estabelecer uma conexão segura com um cliente de jogo e para criptografar a comunicação entre cliente e servidor.

Sintaxe

```
GetInstanceCertificateOutcome GetInstanceCertificate();
```

Parâmetros

Essa ação não tem um parâmetro.

Valor de retorno

Se for bem-sucedido, o retorno será um objeto `GetInstanceCertificateOutcome` que contém a localização do arquivo de certificado TLS e da cadeia de certificados da frota, que estão armazenados na instância. Um arquivo de certificado raiz, extraído da cadeia de certificados, também é armazenado na instância. Se não for bem-sucedido, retornará uma mensagem de erro.

Para obter mais informações sobre o certificado e os dados da cadeia de certificados, consulte [Elementos de GetCertificate resposta](#) na Referência da AWS Certificate Manager API.

Exemplo

```
var getInstanceCertificateOutcome = GameLiftServerAPI.GetInstanceCertificate();
```

GetSdkVersion()

Retorna o número da versão atual do SDK compilado no processo de servidor.

Sintaxe

```
AwsStringOutcome GetSdkVersion()
```

Parâmetros

Essa ação não tem um parâmetro.

Valor de retorno

Se bem-sucedido, retornará a versão do SDK atual como um objeto `AwsStringOutcome`. A string retornada inclui apenas o número da versão (exemplo, "3.1.5"). Se não for bem-sucedido, retornará uma mensagem de erro.

Exemplo

```
var getSdkVersionOutcome = GameLiftServerAPI.GetSdkVersion();
```

GetTerminationTime()

Retorna a hora em que um processo do servidor está programado para ser desligado, se essa informação estiver disponível. Um processo do servidor executa essa ação depois de receber um `onProcessTerminate()` retorno de chamada do Amazon GameLift Servers serviço. Amazon GameLift Servers [pode ligar `onProcessTerminate\(\)` pelos seguintes motivos: \(1\) por problemas de saúde \(o processo do servidor relatou a integridade da porta ou não respondeu\) Amazon GameLift Servers, \(2\) ao encerrar a instância durante um evento de redução ou \(3\) quando uma instância está sendo encerrada devido a uma interrupção da instância spot.](#)

Se o processo tiver recebido um retorno de chamada `onProcessTerminate()`, o valor retornado será o tempo estimado de encerramento. Se o processo não tiver recebido um retorno de chamada `onProcessTerminate()`, uma mensagem de erro será retornada. Saiba mais sobre [como desligar um processo do servidor](#).

Sintaxe

```
AwsDateTimeOutcome GetTerminationTime()
```

Parâmetros

Essa ação não tem um parâmetro.

Valor de retorno

Se for bem-sucedido, retornará o horário de término como um objeto `AwsDateTimeOutcome`. O valor é o tempo de término, expresso em tiques decorridos desde 0001 00:00:00. Por exemplo, o valor da data e hora 2020-09-13 12:26:40 -000Z é igual aos tiques 637355968000000000. Se nenhum horário de rescisão estiver disponível, o retornará uma mensagem de erro.

Exemplo

```
var getTerminationTimeOutcome = GameLiftServerAPI.GetTerminationTime();
```

InitSDK()

Inicializa o Amazon GameLift Servers SDK. Esse método deve ser chamado na inicialização antes de qualquer outra inicialização relacionada ao Amazon GameLift Servers.

Sintaxe

```
InitSDKOutcome InitSDK()
```

Parâmetros

Essa ação não tem um parâmetro.

Valor de retorno

Se for bem-sucedido, retornará um `InitSdkOutcome` objeto indicando que o processo do servidor está pronto para ser chamado [ProcessReady\(\)](#).

Exemplo

```
var initSDKOutcome = GameLiftServerAPI.InitSDK();
```

ProcessEnding()

Notifica o serviço Amazon GameLift Servers de que o processo do servidor está sendo desligado. Esse método deverá ser chamado depois de todas as outras tarefas de limpeza, inclusive desligar todas as sessões de jogos ativas. Este método deve sair com um código de saída zero; um código de saída diferente de zero resulta em uma mensagem de evento indicando que o processo não foi encerrado corretamente.

Depois que o método sair com um código de 0, você poderá encerrar o processo com um código de saída bem-sucedido. Você também poderá sair do processo com um código de erro. Se você sair com um código de erro, o evento da frota indicará que o processo foi encerrado de forma anormal (`SERVER_PROCESS_TERMINATED_UNHEALTHY`).

Sintaxe

```
GenericOutcome ProcessEnding()
```

Parâmetros

Essa ação não tem um parâmetro.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

```
var processEndingOutcome = GameLiftServerAPI.ProcessEnding();
if (processReadyOutcome.Success)
    Environment.Exit(0);
// otherwise, exit with error code
Environment.Exit(errorCode);
```

ProcessReady()

Notifica o serviço Amazon GameLift Servers de que o processo do servidor está pronto para hospedar sessões do jogo. Chame esse método depois de invocar [InitSDK\(\)](#) e concluir todas as tarefas de configuração necessárias antes que o processo do servidor possa hospedar uma sessão do jogo. Esse método deve ser chamado somente uma vez por processo.

Sintaxe

```
GenericOutcome ProcessReady(ProcessParameters processParameters)
```

Parâmetros

processParameters

Um objeto [ProcessParameters](#) que fornece as seguintes informações sobre o processo do servidor:

- Nomes de métodos de retorno de chamada, implementados no código do servidor de jogos, que o serviço Amazon GameLift Servers invoca para se comunicar com o processo de servidor.
- Número da porta em que o processo de servidor está escutando.

- Caminho de qualquer arquivo específico da sessão do jogo que você deseja que o Amazon GameLift Servers capture e armazene.

Obrigatório: Sim

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

Este exemplo ilustra as implementações das funções de chamada e delegação [ProcessReady\(\)](#).

```
// 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"
        })
    );

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 o serviço Amazon GameLift Servers de que um jogador com o ID da sessão do jogador especificado se desconectou do processo do servidor. Em resposta, o Amazon GameLift Servers altera o slot do jogador para um disponível, o que permite ser atribuído a um novo jogador.

Sintaxe

```
GenericOutcome RemovePlayerSession(String playerId)
```

Parâmetros

playerSessionId

ID exclusivo emitido Amazon GameLift Servers quando uma nova sessão de jogador é criada. O ID da sessão do jogador é especificado em um `PlayerSession` objeto, que é retornado em resposta a uma chamada do cliente para as ações da GameLift API [StartGameSessionPlacement CreateGameSession](#), [DescribeGameSessionPlacement](#), ou [DescribePlayerSessions](#).

Tipo: string

Obrigatório: Sim

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

```
Aws::GameLift::GenericOutcome disconnectOutcome =
    Aws::GameLift::Server::RemovePlayerSession(playerSessionId);
```

StartMatchBackfill()

Envia uma solicitação para encontrar novos jogadores para os slots abertos em uma sessão de jogo criada com o FlexMatch. Veja também a ação do AWS SDK [StartMatchBackfill\(\)](#). Com essa ação, as solicitações de alocação de correspondência podem ser iniciadas por um processo do servidor de jogos que esteja hospedando a sessão do jogo. Saiba mais sobre o [recurso FlexMatch de preenchimento](#).

Esta ação é assíncrona. Se a correspondência dos novos jogadores for bem-sucedida, o serviço Amazon GameLift Servers entregará os dados atualizados do marcador de jogos usando a função de retorno de chamada `OnUpdateGameSession()`.

Um processo de servidor pode ter apenas uma solicitação de alocação de correspondência ativa por vez. Para enviar uma nova solicitação, primeiro chame [StopMatchBackfill\(\)](#) para cancelar a solicitação original.

Sintaxe

```
StartMatchBackfillOutcome StartMatchBackfill (StartMatchBackfillRequest  
startBackfillRequest);
```

Parâmetros

StartMatchBackfillRequest

Um objeto [StartMatchBackfillRequest](#) que fornece as seguintes informações:

- ID do tíquete a ser atribuído à solicitação de alocação. Essa informação é opcional. Caso nenhum ID seja fornecido, o Amazon GameLift Servers gerará um automaticamente.
- O marcador de jogos para o qual a solicitação é enviada. O ARN completo da configuração é necessário. Esse valor pode ser obtido dos dados do marcador da sessão do jogo.
- ID da sessão de jogo que está sendo alocada.
- Dados disponíveis de marcação para os jogadores atuais da sessão do jogo.

Obrigatório: Sim

Valor de retorno

Retorna um `StartMatchBackfillOutcome` objeto com o ID do ticket de preenchimento correspondente ou falha com uma mensagem de erro.

Exemplo

```
// 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()

Cancela uma solicitação de alocação de correspondência ativa criada com [StartMatchBackfill\(\)](#). Veja também a ação do AWS SDK [StopMatchmaking\(\)](#). Saiba mais sobre o [recurso FlexMatch de preenchimento](#).

Sintaxe

```
GenericOutcome StopMatchBackfill (StopMatchBackfillRequest stopBackfillRequest);
```

Parâmetros

StopMatchBackfillRequest

Um objeto [StopMatchBackfillRequest](#) que identifica o tíquete de marcação de jogos para cancelar:

- ID do tíquete atribuído à solicitação de alocação sendo cancelada
- o marcador de jogo que recebeu a solicitação de alocação
- a sessão do jogo associada à solicitação de alocação

Obrigatório: Sim

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

```
// 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()

Esse método foi descontinuado com a versão 4.0.1. Em vez disso, o processo do servidor deve ser chamado [ProcessEnding\(\)](#) após o término de uma sessão de jogo.

Notifica o Amazon GameLift Servers serviço de que o processo do servidor encerrou a sessão atual do jogo. Essa ação é chamada quando o processo do servidor permanece ativo e pronto para hospedar uma nova sessão de jogo. Ele deve ser chamado somente após a conclusão do

procedimento de encerramento da sessão de jogo, pois indica Amazon GameLift Servers que o processo do servidor está imediatamente disponível para hospedar uma nova sessão de jogo.

Essa ação não será chamada se o processo do servidor for encerrado após o término da sessão do jogo. Em vez disso, chame [ProcessEnding\(\)](#) para sinalizar que tanto a sessão do jogo quanto o processo do servidor estão terminando.

Sintaxe

```
GenericOutcome TerminateGameSession()
```

Parâmetros

Essa ação não tem um parâmetro.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

Este exemplo ilustra um processo de servidor no final de uma sessão de jogo.

```
// 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()

Atualiza a capacidade da sessão do jogo atual para aceitar novas sessões de jogador. Atualiza a capacidade da sessão do jogo atual para aceitar novas sessões de jogador. (Veja também a ação [UpdateGameSession\(\)](#) na Referência da API Amazon GameLift Servers de serviço).

Sintaxe

```
GenericOutcome UpdatePlayerSessionCreationPolicy(PlayerSessionCreationPolicy  
playerSessionPolicy)
```

Parâmetros

newPlayerSessionPolítica

Valor de string que indica se a sessão do jogo aceita novos jogadores.

Tipo: enumeração [PlayerSessionCreationPolicy](#). Os valores válidos são:

- ACCEPT_ALL – Aceite todas as novas sessões de jogador.
- DENY_ALL – Recuse todas as novas sessões de jogador.

Obrigatório: Sim

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

Este exemplo define a política de ingresso da sessão do jogo atual para aceitar todos os jogadores.

```
var updatePlayerSessionCreationPolicyOutcome =  
  
    GameLiftServerAPI.UpdatePlayerSessionCreationPolicy(PlayerSessionCreationPolicy.ACCEPT_ALL);
```

SDK do servidor (Unreal) para Amazon GameLift Servers -- Ações

Use o SDK do servidor para Unreal para integrar seu jogo multijogador para hospedagem com Amazon GameLift Servers. Para obter orientação sobre o processo de integração, consulte [Adicione Amazon GameLift Servers ao seu servidor de jogos](#).

Note

Essa referência é para uma versão anterior do SDK do servidor para Amazon GameLift Servers. Para obter a versão mais recente, consulte [SDK 5.x do servidor C++ \(Unreal\) para -- Ações Amazon GameLift Servers](#).

Esta API é definida em `GameLiftServerSDK.h` e `GameLiftServerSDKModels.h`.

Para configurar o plug-in do Unreal Engine e ver exemplos de código [Amazon GameLift ServersIntegre-se a um projeto Unreal Engine](#).

SDK do servidor (Unreal) para Amazon GameLift Servers -- Tipos de dados

Use o comando Amazon GameLift Servers Referência do SDK do servidor para Unreal para integrar seu jogo multijogador para hospedagem com Amazon GameLift Servers. Para obter orientação sobre o processo de integração, consulte[Adicione Amazon GameLift Servers ao seu servidor de jogos](#).

Note

Essa referência é para uma versão anterior do SDK do servidor para Amazon GameLift Servers. Para obter a versão mais recente, consulte[SDK 5.x do servidor C++ \(Unreal\) para Amazon GameLift Servers -- Tipos de dados](#).

Esta API é definida em `GameLiftServerSDK.h` e `GameLiftServerSDKModels.h`.

Para configurar o plug-in do Unreal Engine e ver exemplos de código [Amazon GameLift ServersIntegre-se a um projeto Unreal Engine](#).

[SDK do servidor \(Unreal\) para Amazon GameLift Servers -- Ações](#)

Tópicos

- [FDescribePlayerSessionsRequest](#)
- [FProcessParâmetros](#)
- [FStartMatchBackfillRequest](#)
- [FStopMatchBackfillRequest](#)

FDescribePlayerSessionsRequest

Esse tipo de dados é usado para especificar quais sessões de jogador recuperar. Você pode usá-lo da seguinte forma:

- Forneça um `PlayerSessionId` para solicitar uma sessão de jogador específica.
- Forneça um `GameSessionId` para solicitar todas as sessões de jogadores na sessão de jogo especificada.
- Forneça um `PlayerId` para solicitar todas as sessões de jogadores para o jogador especificado.

Para grandes coleções de sessões de jogador, use os parâmetros de paginação para recuperar resultados em blocos sequenciais.

Conteúdo

GameSessionId

Identificador da sessão de jogo exclusivo. Use esse parâmetro para solicitar todas as sessões de jogador da sessão de jogo especificada. O formato do ID da sessão de jogo é o seguinte: `arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string>`. O valor de `<ID string>` é uma string de ID personalizada ou (se houver uma especificada quando a sessão de jogo foi criada) uma string gerada.

Tipo: String

Obrigatório: Não

Limite

O número máximo de resultados a serem retornados. Use esse parâmetro com `NextToken` para obter resultados como um conjunto de páginas sequenciais. Se um ID de sessão de jogador for especificado, esse parâmetro será ignorado.

Tipo: número inteiro

Obrigatório: não

NextToken

Token indicando o início da próxima página sequencial de resultados. Use o token retornado com uma chamada anterior para essa ação. Para especificar o início do conjunto de resultados, não especifique um valor. Se um ID de sessão de jogador for especificado, esse parâmetro será ignorado.

Tipo: String

Obrigatório: Não

PlayerId

Identificador exclusivo para um jogador. IDs Os jogadores são definidos pelo desenvolvedor. Consulte [Gerar jogador IDs](#).

Tipo: String

Obrigatório: Não

PlayerSessionId

Identificador exclusivo para uma sessão de jogador.

Tipo: String

Obrigatório: Não

PlayerSessionStatusFilter

Status da sessão de jogador para filtrar resultados. Entre os status da sessão de jogador possíveis estão os seguintes:

- RESERVED – A solicitação de sessão de jogador foi recebida, mas o jogador ainda não se conectou ao processo de servidor e/ou foi validado.
- ACTIVE – O jogador foi validado pelo processo de servidor e está conectado no momento.
- COMPLETED – A conexão do jogador foi interrompida.
- TIMEDOUT – Uma solicitação de sessão de jogador foi recebida, mas o jogador não se conectou e/ou não foi validado dentro do tempo limite (60 segundos).

Tipo: String

Obrigatório: Não

FProcessParâmetros

Esse tipo de dados contém o conjunto de parâmetros enviados para o Amazon GameLift Servers serviço em uma [ProcessReady\(\)](#) chamada.

Conteúdo

porta

Número da porta em que o processo de servidor escutará novas conexões de jogador. O valor deve estar dentro do intervalo de portas configurado para qualquer frota que implanta essa compilação do servidor de jogo. Esse número de porta está incluído nos objetos das sessões de jogo e jogador, que as sessões de jogo usam ao se conectar a um processo de servidor.

Tipo: inteiro

Obrigatório: sim

logParameters

Objeto com uma lista de caminhos de diretório para arquivos de log da sessão de jogo.

Tipo: TArray < FString >

Obrigatório: não

onStartGameSessão

Nome da função de retorno de chamada que o Amazon GameLift Servers o serviço é invocado para ativar uma nova sessão de jogo. Amazon GameLift Servers chama essa função em resposta à solicitação do cliente [CreateGameSession](#). A função de retorno de chamada usa um [GameSession](#) objeto (definido no Amazon GameLift Servers Referência da API de serviço).

Tipo: FOnStartGameSession

Obrigatório: sim

onProcessTerminate

Nome da função de retorno de chamada que o Amazon GameLift Servers o serviço é invocado para forçar o encerramento do processo do servidor. Depois de chamar essa função, Amazon GameLift Servers espera cinco minutos até que o processo do servidor seja encerrado e responda com uma [ProcessEnding\(\)](#) chamada antes de encerrar o processo do servidor.

Tipo: FSimple Delegado

Obrigatório: não

onHealthCheck

Nome da função de retorno de chamada que o Amazon GameLift Servers o serviço é chamado para solicitar um relatório de status de saúde do processo do servidor. Amazon GameLift Servers chama essa função a cada 60 segundos. Depois de chamar esta função Amazon GameLift Servers espera 60 segundos por uma resposta e, se nenhuma for recebida, registra o processo do servidor como não íntegro.

Tipo: FOnHealthCheck

Obrigatório: não

onUpdateGameSession

Nome da função de retorno de chamada que o Amazon GameLift Servers o serviço é invocado para passar um objeto de sessão de jogo atualizado para o processo do servidor. Amazon GameLift Servers chama essa função quando uma solicitação de [preenchimento de correspondência](#) é processada para fornecer dados atualizados do matchmaker. Ele transmite um objeto [GameSession](#), uma atualização de status (updateReason) e o ID do tíquete de alocação de correspondência.

Tipo: FOnUpdateGameSession

Obrigatório: não

FStartMatchBackfillRequest

Esse tipo de dados é usado para enviar uma solicitação de alocação de marcação de jogos. As informações são comunicadas ao Amazon GameLift Servers serviço em uma [StartMatchBackfill\(\)](#) chamada.

Conteúdo

GameSessionArn

Identificador da sessão de jogo exclusivo. A ação de API [GetGameSessionId\(\)](#) retorna o identificador no formato de ARN.

Tipo: FString

Obrigatório: sim

MatchmakingConfigurationArn

O identificador exclusivo, no formato de ARN, para o marcador de jogos a ser usado para essa solicitação. Para encontrar o marcador de jogos usado para criar a sessão de jogo original, procure no objeto de sessão do jogo, na propriedade de dados do marcador de jogos. Saiba mais sobre os dados de marcador de jogos em [Trabalhar com dados do marcador de jogos](#).

Tipo: FString

Obrigatório: sim

Jogadores

Um conjunto de dados que representa todos os jogadores que estão na sessão do jogo no momento. O marcador de jogos usa essas informações para pesquisar novos jogadores que sejam correspondências ideais para os jogadores atuais. Veja o Amazon GameLift Servers Guia de referência da API para uma descrição do formato do objeto Player. Para encontrar os atributos do jogador e as atribuições da equipe, procure no objeto da sessão do jogo, na propriedade de dados do matchmaker. IDs Se o marcador de jogos usa latência, obtenha a latência atualizada para a região atual e a inclua nos dados de cada jogador.

Tipo: TArray [< FPlayer >](#)

Obrigatório: sim

TicketId

O identificador exclusivo de um tíquete de solicitação de marcação de jogo ou de alocação de correspondência. Se nenhum valor for fornecido aqui, Amazon GameLift Servers gerará um na forma de um UUID. Use esse identificador para acompanhar o status do tíquete de alocação de correspondência ou cancelar a solicitação, se necessário.

Tipo: FString

Obrigatório: não

FStopMatchBackfillRequest

Esse tipo de dados é usado para cancelar uma solicitação de alocação de marcação de jogo. As informações são comunicadas ao Amazon GameLift Servers serviço em uma [StopMatchBackfill\(\)](#) chamada.

Conteúdo

GameSessionArn

O identificador de sessão de jogo exclusivo associado à solicitação que está sendo cancelada.

Tipo: FString

Obrigatório: sim

MatchmakingConfigurationArn

O identificador exclusivo do marcador de jogos para o qual essa solicitação foi enviada.

Tipo: FString

Obrigatório: sim

TicketId

O identificador exclusivo do tíquete de solicitação de alocação a ser cancelado.

Tipo: FString

Obrigatório: Sim

[SDK do servidor \(Unreal\) para Amazon GameLift Servers -- Tipos de dados](#)

Tópicos

- [AcceptPlayerSession\(\)](#)
- [ActivateGameSession\(\)](#)
- [DescribePlayerSessions\(\)](#)
- [GetGameSessionId\(\)](#)
- [GetInstanceCertificate\(\)](#)
- [GetSdkVersion\(\)](#)
- [InitSDK\(\)](#)
- [ProcessEnding\(\)](#)
- [ProcessReady\(\)](#)
- [RemovePlayerSession\(\)](#)
- [StartMatchBackfill\(\)](#)
- [StopMatchBackfill\(\)](#)
- [TerminateGameSession\(\)](#)
- [UpdatePlayerSessionCreationPolicy\(\)](#)

AcceptPlayerSession()

Notifica o Amazon GameLift Servers serviço em que um jogador com o ID de sessão de jogador especificado conectou ao processo do servidor e precisa de validação. Amazon GameLift Servers verifica se o ID da sessão do jogador é válido, ou seja, se o ID do jogador reservou um espaço para

o jogador na sessão do jogo. Uma vez validado, Amazon GameLift Servers altera o status da vaga do jogador de RESERVADO para ATIVO.

Sintaxe

```
FGameLiftGenericOutcome AcceptPlayerSession(const FString& playerId)
```

Parâmetros

playerSessionId

ID exclusivo emitido pelo Amazon GameLift Servers serviço em resposta a uma chamada para o AWS SDK Amazon GameLift Servers Ação da API [CreatePlayerSession](#). O cliente do jogo faz referência a esse ID ao se conectar ao processo do servidor.

Tipo: FString

Obrigatório: sim

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

ActivateGameSession()

Notifica o Amazon GameLift Servers serviço em que o processo do servidor ativou uma sessão de jogo e agora está pronto para receber conexões de jogadores. Essa ação deve ser chamada como parte da função de retorno de chamada `onStartGameSession()`, depois que toda a inicialização da sessão tiver sido concluída.

Sintaxe

```
FGameLiftGenericOutcome ActivateGameSession()
```

Parâmetros

Essa ação não tem um parâmetro.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

DescribePlayerSessions()

Recupera dados da sessão do jogador, inclusive configurações, metadados da sessão e dados do jogador. Use essa ação para obter informações de uma única sessão de jogador, para todas as sessões de jogador em uma sessão de jogo, ou para todas as sessões de jogador associadas a um único ID de jogador.

Sintaxe

```
FGameLiftDescribePlayerSessionsOutcome DescribePlayerSessions(const  
    FGameLiftDescribePlayerSessionsRequest &describePlayerSessionsRequest)
```

Parâmetros

describePlayerSessionsSolicitação

Um objeto [FDescribePlayerSessionsRequest](#) que descreve quais sessões de jogador recuperar.

Obrigatório: sim

Valor de retorno

Se bem-sucedido, retorna um objeto [FDescribePlayerSessionsRequest](#) que contém um conjunto de objetos de sessão do jogador que atendem aos parâmetros de solicitação. Os objetos da sessão do jogador têm uma estrutura idêntica à do AWS SDK Amazon GameLift Servers Tipo [PlayerSession](#) de dados da API.

GetGameSessionId()

Recupera o ID da sessão do jogo hospedada no momento pelo processo do servidor, caso o processo do servidor esteja ativo.

Sintaxe

```
FGameLiftStringOutcome GetGameSessionId()
```

Parâmetros

Essa ação não tem um parâmetro.

Valor de retorno

Se bem-sucedido, retornará o ID da sessão de jogo como um objeto `FGameLiftStringOutcome`. Se não for bem-sucedido, retornará uma mensagem de erro.

GetInstanceCertificate()

Recupera a localização do arquivo de um certificado TLS codificado por pem que está associado à frota e suas instâncias. AWS Certificate Manager gera esse certificado quando você cria uma nova frota com a configuração do certificado definida como `GENERATED`. Use esse certificado para estabelecer uma conexão segura com um cliente de jogo e para criptografar a comunicação entre cliente e servidor.

Sintaxe

```
FGameLiftGetInstanceCertificateOutcome GetInstanceCertificate()
```

Parâmetros

Essa ação não tem um parâmetro.

Valor de retorno

Se for bem-sucedido, o retorno será um objeto `GetInstanceCertificateOutcome` que contém a localização do arquivo de certificado TLS e da cadeia de certificados da frota, que estão armazenados na instância. Um arquivo de certificado raiz, extraído da cadeia de certificados, também é armazenado na instância. Se não for bem-sucedido, retornará uma mensagem de erro.

Para obter mais informações sobre o certificado e os dados da cadeia de certificados, consulte [Elementos de GetCertificate resposta](#) na Referência da AWS Certificate Manager API.

GetSdkVersion()

Retorna o número da versão atual do SDK compilado no processo de servidor.

Sintaxe

```
FGameLiftStringOutcome GetSdkVersion();
```

Parâmetros

Essa ação não tem um parâmetro.

Valor de retorno

Se bem-sucedido, retornará a versão do SDK atual como um objeto `FGameLiftStringOutcome`. A string retornada inclui apenas o número da versão (exemplo, "3.1.5"). Se não for bem-sucedido, retornará uma mensagem de erro.

Exemplo

```
Aws::GameLift::AwsStringOutcome SdkVersionOutcome =  
    Aws::GameLift::Server::GetSdkVersion();
```

InitSDK()

Inicializa o Amazon GameLift Servers SDK. Esse método deve ser chamado no lançamento, antes de qualquer outro Amazon GameLift Serversa inicialização relacionada ocorre.

Sintaxe

```
FGameLiftGenericOutcome InitSDK()
```

Parâmetros

Essa ação não tem um parâmetro.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

ProcessEnding()

Notifica o Amazon GameLift Servers serviço que o processo do servidor está encerrando. Esse método deverá ser chamado depois de todas as outras tarefas de limpeza, inclusive desligar todas as sessões de jogos ativas. Este método deve sair com um código de saída zero; um código de saída diferente de zero resulta em uma mensagem de evento indicando que o processo não foi encerrado corretamente.

Sintaxe

```
FGameLiftGenericOutcome ProcessEnding()
```

Parâmetros

Essa ação não tem um parâmetro.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

ProcessReady()

Notifica o Amazon GameLift Servers serviço em que o processo do servidor está pronto para hospedar sessões de jogo. Chame esse método depois de invocar [InitSDK\(\)](#) e concluir todas as tarefas de configuração necessárias antes que o processo do servidor possa hospedar uma sessão do jogo. Esse método deve ser chamado somente uma vez por processo.

Sintaxe

```
FGameLiftGenericOutcome ProcessReady(FProcessParameters &processParameters)
```

Parâmetros

FProcessParâmetros

Um objeto [FProcessParâmetros](#) que fornece as seguintes informações sobre o processo do servidor:

- Nomes dos métodos de retorno de chamada, implementados no código do servidor do jogo, que o Amazon GameLift Servers o serviço é invocado para se comunicar com o processo do servidor.
- Número da porta em que o processo de servidor está escutando.
- Caminho para qualquer arquivo específico da sessão de jogo que você deseja Amazon GameLift Servers para capturar e armazenar.

Obrigatório: sim

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Exemplo

Consulte o código de exemplo em [Usar o plug-in do Unreal Engine](#).

RemovePlayerSession()

Notifica o Amazon GameLift Servers serviço que um jogador com o ID de sessão de jogador especificado desconectou do processo do servidor. Em resposta, Amazon GameLift Servers altera o espaço do jogador para disponível, o que permite que ele seja atribuído a um novo jogador.

Sintaxe

```
FGameLiftGenericOutcome RemovePlayerSession(const FString& playerId)
```

Parâmetros

playerSessionId

ID exclusivo emitido pelo Amazon GameLift Servers serviço em resposta a uma chamada para o AWS SDK Amazon GameLift Servers Ação da API [CreatePlayerSession](#). O cliente do jogo faz referência a esse ID ao se conectar ao processo do servidor.

Tipo: FString

Obrigatório: sim

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

StartMatchBackfill()

Envia uma solicitação para encontrar novos jogadores para vagas abertas em uma sessão de jogo criada com FlexMatch. Veja também a ação do AWS SDK [StartMatchBackfill\(\)](#). Com essa ação, as solicitações de alocação de correspondência podem ser iniciadas por um processo do servidor de jogos que esteja hospedando a sessão do jogo. Saiba mais sobre o [FlexMatch recurso](#) de preenchimento.

Esta ação é assíncrona. Se novos jogadores forem combinados com sucesso, o Amazon GameLift Servers o serviço fornece dados atualizados do matchmaker usando a função de retorno de chamada. `OnUpdateGameSession()`

Um processo de servidor pode ter apenas uma solicitação de alocação de correspondência ativa por vez. Para enviar uma nova solicitação, primeiro chame [StopMatchBackfill\(\)](#) para cancelar a solicitação original.

Sintaxe

```
FGameLiftStringOutcome StartMatchBackfill (FStartMatchBackfillRequest  
&startBackfillRequest);
```

Parâmetros

FStartMatchBackfillRequest

Um objeto [FStartMatchBackfillRequest](#) que fornece as seguintes informações:

- ID do tíquete a ser atribuído à solicitação de alocação. Essas informações são opcionais; se nenhum ID for fornecido, Amazon GameLift Servers gerará automaticamente um.
- O marcador de jogos para o qual a solicitação é enviada. O ARN completo da configuração é necessário. Esse valor pode ser obtido dos dados do marcador da sessão do jogo.
- ID da sessão de jogo que está sendo alocada.
- Dados disponíveis de marcação para os jogadores atuais da sessão do jogo.

Obrigatório: sim

Valor de retorno

Se bem-sucedido, retorna o tíquete de alocação de correspondência como um objeto `FGameLiftStringOutcome`. Se não for bem-sucedido, retornará uma mensagem de erro. O status do ticket pode ser monitorado usando a ação do AWS SDK [DescribeMatchmaking\(\)](#).

StopMatchBackfill()

Cancela uma solicitação de alocação de correspondência ativa criada com [StartMatchBackfill\(\)](#). Veja também a ação do AWS SDK [StopMatchmaking\(\)](#). Saiba mais sobre o [FlexMatch recurso](#) de preenchimento.

Sintaxe

```
FGameLiftGenericOutcome StopMatchBackfill (FStopMatchBackfillRequest  
&stopBackfillRequest);
```

Parâmetros

StopMatchBackfillRequest

Um objeto [FStopMatchBackfillRequest](#) que identifica o tíquete de marcação de jogos para cancelar:

- ID do tíquete atribuído à solicitação de alocação sendo cancelada
- o marcador de jogo que recebeu a solicitação de alocação
- a sessão do jogo associada à solicitação de alocação

Obrigatório: sim

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

TerminateGameSession()

Esse método foi descontinuado com a versão 4.0.1. Em vez disso, o processo do servidor deve ser chamado [ProcessEnding\(\)](#) após o término de uma sessão de jogo.

Notifica o Amazon GameLift Servers serviço de que o processo do servidor encerrou a sessão atual do jogo. Essa ação é chamada quando o processo do servidor permanece ativo e pronto para hospedar uma nova sessão de jogo. Ele deve ser chamado somente após a conclusão do procedimento de encerramento da sessão de jogo, pois sinaliza para Amazon GameLift Servers que o processo do servidor esteja imediatamente disponível para hospedar uma nova sessão de jogo.

Essa ação não será chamada se o processo do servidor for encerrado após o término da sessão do jogo. Em vez disso, chame [ProcessEnding\(\)](#) para sinalizar que tanto a sessão do jogo quanto o processo do servidor estão terminando.

Sintaxe

```
FGameLiftGenericOutcome TerminateGameSession()
```

Parâmetros

Essa ação não tem um parâmetro.

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

UpdatePlayerSessionCreationPolicy()

Atualiza a capacidade da sessão do jogo atual para aceitar novas sessões de jogador. Atualiza a capacidade da sessão do jogo atual para aceitar novas sessões de jogador. (Veja também a [UpdateGameSession\(\)](#) ação no Amazon GameLift Servers Referência da API de serviço).

Sintaxe

```
FGameLiftGenericOutcome UpdatePlayerSessionCreationPolicy(EPlayerSessionCreationPolicy policy)
```

Parâmetros

Política

Valor indicando se a sessão do jogo aceita novos jogadores.

Tipo: enumeração EPlayerSessionCreationPolicy. Os valores válidos são:

- ACCEPT_ALL – Aceite todas as novas sessões de jogador.
- DENY_ALL – Recuse todas as novas sessões de jogador.

Obrigatório: sim

Valor de retorno

Retorna um resultado genérico que consiste em sucesso ou falha com uma mensagem de erro.

Amazon GameLift ServersAmazon GameLift ServersreferênciaRealtime

Esta seção contém documentação de referência para o Amazon GameLift Servers Amazon GameLift Servers Realtime SDK. Ele inclui a API Realtime do cliente, bem como orientações para configurar seu Amazon GameLift Servers Realtime script.

Tópicos

- [Amazon GameLift ServersRealtimereferência da API do cliente \(C#\)](#)

- [Referência de script para Amazon GameLift ServersRealtime](#)

Amazon GameLift ServersRealtime referência da API do cliente (C#)

Use a API Realtime do cliente para preparar seus clientes de jogos multijogador para uso com Amazon GameLift Servers Amazon GameLift ServersRealtime. A API do cliente contém um conjunto de chamadas de API síncronas e retornos de chamada assíncronos que permitem que um cliente do jogo se conecte a um servidor Realtime e troque mensagens e dados com outros clientes de jogos por meio do servidor.

A API é definida nas seguintes bibliotecas:

Client.cs

- [Ações síncronas](#)
- [Retornos de chamada assíncronos](#)
- [Tipos de dados](#)

Para configurar a API Realtime do cliente

1. Baixe o [SDK Amazon GameLift ServersRealtime do cliente](#).
2. Compile as bibliotecas do C# SDK. Localize o arquivo de solução `GameLiftRealtimeClientSdkNet45.sln`. Consulte o arquivo do `README.md` para o SDK de servidor para obter os requisitos mínimos e as opções de compilação adicionais. Em um IDE, carregue o arquivo da solução. Para gerar as bibliotecas do SDK, restaure os NuGet pacotes e crie a solução.
3. Adicione as bibliotecas de cliente do Realtime ao projeto do cliente de jogo.

Amazon GameLift ServersRealtime referência da API do cliente (C#): Ações

Essa referência da API Realtime do cliente C# pode ajudá-lo a preparar seu jogo multijogador para uso em Amazon GameLift Servers Realtime Amazon GameLift Servers frotas.

- [Ações síncronas](#)
- [Retornos de chamada assíncronos](#)
- [Tipos de dados](#)

Client()

Inicializa um novo cliente para se comunicar com o servidor em tempo real e identifica o tipo de conexão a ser usado.

Sintaxe

```
public Client(ClientConfiguration configuration)
```

Parâmetros

clientConfiguration

Detalhes de configuração especificando o tipo de client/server conexão. Você pode optar por chamar Client() sem esse parâmetro; no entanto, essa abordagem resulta em uma conexão não segura por padrão.

Tipo: [ClientConfiguration](#)

Obrigatório: Não

Valor de retorno

Retorna uma instância do cliente Realtime para uso com a comunicação com o servidor Realtime.

Connect()

Solicita uma conexão com um processo de servidor que está hospedando uma sessão do jogo.

Sintaxe

```
public ConnectionStatus Connect(string endpoint, int remoteTcpPort, int listenPort,
    ConnectionToken token)
```

Parâmetros

endpoint

Nome DNS ou endereço IP da sessão de jogo à qual se conectar. O endpoint é especificado em um GameSession objeto, que é retornado em resposta a uma chamada do cliente para as ações da Amazon GameLift ServersAPI do AWS SDK [StartGameSessionPlacementCreateGameSession](#), ou. [DescribeGameSessions](#)

 Note

Se o servidor Realtime estiver em execução em uma frota com um certificado TLS, você deverá usar o nome DNS.

Tipo: string

Obrigatório: Sim

remoteTcpPort

Número da porta da conexão TCP atribuída à sessão do jogo. Essas informações são especificadas em um `GameSession` objeto, que é retornado em resposta a uma [StartGameSessionPlacementCreateGameSessionDescribeGameSession](#) solicitação ou.

Tipo: inteiro

Valores válidos: 1900 a 2000.

Obrigatório: Sim

listenPort

Número da porta em que o cliente de jogo está escutando mensagens enviadas usando o canal UDP.

Tipo: inteiro

Valores válidos: 33400 a 33500.

Obrigatório: Sim

token

Informações opcionais que identificam o cliente do jogo solicitante para o processo do servidor.

Tipo: [ConnectionToken](#)

Obrigatório: sim

Valor de retorno

Retorna [ConnectionStatus](#) um valor enumerado indicando o status da conexão do cliente.

Disconnect()

Quando conectado a uma sessão de jogo, desconecta o cliente do jogo da sessão do jogo.

Sintaxe

```
public void Disconnect()
```

Parâmetros

Essa ação não tem um parâmetro.

Valor de retorno

Esse método não retorna nada.

NewMessage()

Cria um novo objeto de mensagem com um código de operação especificado. Assim que um objeto de mensagem for retornado, conclua o conteúdo da mensagem especificando um destino, atualizando o método de entrega e adicionando uma carga útil de dados conforme necessário. Uma vez concluído, envie a mensagem usando `SendMessage()`.

Sintaxe

```
public RTMessage NewMessage(int opCode)
```

Parâmetros

opCode

Código de operação definido pelo desenvolvedor que identifica um evento de jogo ou ação, como uma movimentação do jogador ou uma notificação do servidor.

Tipo: inteiro

Obrigatório: Sim

Valor de retorno

Retorna um objeto [RTMessage](#) que contém o código de operação especificado e o método de entrega padrão. O parâmetro de intenção de entrega é definido como FAST por padrão.

SendMessage()

Envia uma mensagem para um jogador ou grupo usando o método de entrega especificado.

Sintaxe

```
public void SendMessage(RTMessage message)
```

Parâmetros

message

Objeto de mensagem que especifica o destinatário o método de entrega e o conteúdo da mensagem.

Tipo: [RTMessage](#)

Obrigatório: sim

Valor de retorno

Esse método não retorna nada.

JoinGroup()

Adiciona o jogador à associação de um grupo especificado. Os grupos podem conter qualquer um dos jogadores conectados ao jogo. Uma vez associado, o jogador recebe todas as futuras mensagens enviadas ao grupo e pode enviar mensagens para todo o grupo.

Sintaxe

```
public void JoinGroup(int targetGroup)
```

Parâmetros

targetGroup

ID exclusivo que identifica o grupo ao qual adicionar o jogador. IDsOs grupos são definidos pelo desenvolvedor.

Tipo: inteiro

Obrigatório: Sim

Valor de retorno

Esse método não retorna nada. Como essa solicitação é enviada usando o método de entrega confiável (TCP), uma solicitação com falha aciona o retorno de chamada [OnError\(\)](#).

LeaveGroup()

Remove o jogador da associação de um grupo especificado. Assim que não estiver mais no grupo, o jogador não receberá as mensagens enviadas ao grupo e não poderá enviar mensagens para todo o grupo.

Sintaxe

```
public void LeaveGroup(int targetGroup)
```

Parâmetros

targetGroup

ID exclusivo que identifica o grupo do qual remover o jogador. IDsOs grupos são definidos pelo desenvolvedor.

Tipo: inteiro

Obrigatório: Sim

Valor de retorno

Esse método não retorna nada. Como essa solicitação é enviada usando o método de entrega confiável (TCP), uma solicitação com falha aciona o retorno de chamada [OnError\(\)](#).

RequestGroupMembership()

Solicita que uma lista de jogadores no grupo especificado seja enviada para o cliente do jogo. Qualquer jogador pode solicitar essas informações, independentemente de ele ser um membro do

grupo ou não. Em resposta a essa solicitação, a lista de associações é enviada ao cliente por meio de um retorno de chamada [OnGroupMembershipUpdated\(\)](#).

Sintaxe

```
public void RequestGroupMembership(int targetGroup)
```

Parâmetros

targetGroup

ID exclusivo que identifica o grupo do qual obter informações de associação. IDs Os grupos são definidos pelo desenvolvedor.

Tipo: inteiro

Obrigatório: Sim

Valor de retorno

Esse método não retorna nada.

Amazon GameLift ServersRealtimereferência da API do cliente (C#): retornos de chamada assíncronos

Use essa referência da API Realtime do cliente C# para ajudá-lo a preparar seu jogo multijogador para uso em Amazon GameLift Servers Realtime Amazon GameLift Servers frotas.

- [Ações síncronas](#)
- Retornos de chamada assíncronos
- [Tipos de dados](#)

Um cliente de jogo precisa implementar esses métodos de retorno de chamada para responder a eventos. O servidor Realtime chama esses retornos de chamada para enviar informações relacionadas ao jogo para o cliente do jogo. Retornos de chamada dos mesmos eventos também podem ser implementados com a lógica de jogos personalizada no script de servidor Realtime. Consulte [Retornos de chamada de script para Amazon GameLift ServersRealtime](#).

Os métodos de retorno de chamada são definidos em `ClientEvents.cs`.

OnOpen()

Chamado quando o processo do servidor aceita a solicitação de conexão do cliente do jogo e abre uma conexão.

Sintaxe

```
public void OnOpen()
```

Parâmetros

Esse método não aceita parâmetros.

Valor de retorno

Esse método não retorna nada.

OnClose()

Chamado quando o processo do servidor encerra a conexão com o cliente do jogo, como após uma sessão do jogo.

Sintaxe

```
public void OnClose()
```

Parâmetros

Esse método não aceita parâmetros.

Valor de retorno

Esse método não retorna nada.

OnError()

Chamado quando ocorre uma falha para uma solicitação de API do cliente em tempo real. Esse retorno de chamada pode ser personalizado para lidar com uma variedade de erros de conexão.

Sintaxe

```
private void OnError(byte[] args)
```

Parâmetros

Esse método não aceita parâmetros.

Valor de retorno

Esse método não retorna nada.

OnDataReceived()

Chamado quando o cliente do jogo recebe uma mensagem do servidor Realtime. Este é o método principal pelo qual as mensagens e as notificações são recebidas por um cliente do jogo.

Sintaxe

```
public void OnDataReceived(DataReceivedEventArgs dataReceivedEventArgs)
```

Parâmetros

dataReceivedEventArgs

Informações relacionadas à atividade de mensagem.

Tipo: [DataReceivedEventArgs](#)

Obrigatório: sim

Valor de retorno

Esse método não retorna nada.

OnGroupMembershipUpdated()

Chamado quando a associação de um grupo ao qual o jogador pertence foi atualizada. Esse retorno de chamada também é invocado quando um cliente chama RequestGroupMembership.

Sintaxe

```
public void OnGroupMembershipUpdated(GroupMembershipEventArgs groupMembershipEventArgs)
```


Parâmetros

groupMembershipEventArgs

Informações relacionadas à atividade de associação do grupo.

Tipo: [GroupMembershipEventArgs](#)

Obrigatório: sim

Valor de retorno

Esse método não retorna nada.

Amazon GameLift ServersRealtimereferência da API do cliente (C#): tipos de dados

Essa referência da API Realtime do cliente C# pode ajudá-lo a preparar seu jogo multijogador para uso em Amazon GameLift Servers Realtime Amazon GameLift Servers frotas.

- [Ações síncronas](#)
- [Retornos de chamada assíncronos](#)
- Tipos de dados

ClientConfiguration

Informações sobre como o cliente do jogo se conecta a um servidor Realtime.

Conteúdo

ConnectionType

Tipo de client/server conexão a ser usada, segura ou não segura. Se você não especificar um tipo de conexão, o padrão será desprotegido.

Note

Ao se conectar a um servidor em tempo real em uma frota protegida com um certificado TLS, você deve usar o valor RT_OVER_WSS_DTLS_. TLS12

Tipo: um valor `ConnectionType` [enum](#).

Obrigatório: Não

ConnectionToken

Informações sobre o and/or jogador cliente do jogo que está solicitando uma conexão com um Realtime servidor.

Conteúdo

playerSessionId

ID exclusivo emitido Amazon GameLift Servers quando uma nova sessão de jogador é criada. O ID da sessão do jogador é especificado em um `PlayerSession` objeto, que é retornado em resposta a uma chamada do cliente para as ações da GameLift API [StartGameSessionPlacement CreateGameSession](#), [DescribeGameSessionPlacement](#), ou [DescribePlayerSessions](#).

Tipo: string

Obrigatório: Sim

payload

Informações definidas pelo desenvolvedor a serem comunicadas ao servidor Realtime na conexão. Isso inclui quaisquer dados arbitrários que possam ser usados para um mecanismo de login personalizado. Para obter exemplos, uma carga útil pode fornecer informações de autenticação a serem processadas pelo script do servidor Realtime antes de permitir que um cliente se conecte.

Tipo: matriz de bytes

Obrigatório: Não

RTMessage

Conteúdo e informações de entrega de uma mensagem. Uma mensagem deve especificar um jogador ou um grupo de destino.

Conteúdo

opCode

Código de operação definido pelo desenvolvedor que identifica um evento de jogo ou ação, como uma movimentação do jogador ou uma notificação do servidor. O código de operação de uma mensagem fornece o contexto para a carga útil de dados que está sendo fornecida. As mensagens criadas usando `NewMessage()` já têm o código de operação definido, mas ele pode ser alterado a qualquer momento.

Tipo: inteiro

Obrigatório: Sim

targetPlayer

ID exclusivo que identifica o jogador que é o destinatário pretendido da mensagem que está sendo enviada. O destino pode ser o próprio servidor (usando o ID do servidor) ou outro jogador (usando um ID do jogador).

Tipo: inteiro

Obrigatório: não

targetGroup

ID exclusivo que identifica o grupo que é o destinatário pretendido da mensagem que está sendo enviada. IDs Os grupos são definidos pelo desenvolvedor.

Tipo: inteiro

Obrigatório: não

deliveryIntent

Indica se a mensagem deve ser enviada usando a conexão TCP confiável ou o canal UDP rápido. Mensagens criadas usando [NewMessage\(\)](#).

Tipo: `DeliveryIntent` enum

Valores válidos: `FAST` | `RELIABLE`

Obrigatório: Sim

payload

Conteúdo da mensagem. Essas informações são estruturadas conforme necessário para serem processadas pelo cliente do jogo com base no código de operação fornecido. Pode conter dados de estado do jogo ou outras informações que precisem ser transmitidas entre os clientes de jogos ou entre um cliente de jogo e o servidor em tempo real.

Tipo: matriz de bytes

Obrigatório: Não

DataReceivedEventArgs

Os dados fornecidos com um retorno de chamada [OnDataReceived\(\)](#).

Conteúdo

sender (remetente)

ID exclusivo que identifica a entidade (ID do jogador ou ID do servidor) que originou a mensagem.

Tipo: inteiro

Obrigatório: Sim

opCode

Código de operação definido pelo desenvolvedor que identifica um evento de jogo ou ação, como uma movimentação do jogador ou uma notificação do servidor. O código de operação de uma mensagem fornece o contexto para a carga útil de dados que está sendo fornecida.

Tipo: inteiro

Obrigatório: Sim

dados

Conteúdo da mensagem. Essas informações são estruturadas conforme necessário para serem processadas pelo cliente do jogo com base no código de operação fornecido. Pode conter dados de estado do jogo ou outras informações que precisem ser transmitidas entre os clientes de jogos ou entre um cliente de jogo e o servidor em tempo real.

Tipo: matriz de bytes

Obrigatório: Não

GroupMembershipEventArgs

Os dados fornecidos com um retorno de chamada [OnGroupMembershipUpdated\(\)](#).

Conteúdo

sender (remetente)

ID exclusivo que identifica o jogador que solicitou uma atualização da associação do grupo.

Tipo: inteiro

Obrigatório: Sim

opCode

Código de operação definido pelo desenvolvedor que identifica um evento ou uma ação de jogo.

Tipo: inteiro

Obrigatório: Sim

groupId

ID exclusivo que identifica o grupo que é o destinatário pretendido da mensagem que está sendo enviada. IDs Os grupos são definidos pelo desenvolvedor.

Tipo: inteiro

Obrigatório: Sim

playerId

Lista de jogadores IDs que são membros atuais do grupo especificado.

Tipo: matriz de inteiros

Obrigatório: Sim

Enumerações

Os enums definidos para o SDK do cliente Amazon GameLift Servers Realtime são definidos da seguinte forma:

ConnectionStatus

- **CONECTADO** — O cliente do jogo está conectado ao Realtime servidor somente com uma conexão TCP. Todas as mensagens, independentemente da intenção de entrega, são enviadas via TCP.
- **CONNECTED_SEND_FAST** — O cliente do jogo está conectado ao Realtime servidor com uma conexão TCP e uma conexão UDP. No entanto, a capacidade de receber mensagens por meio de UDP ainda não é verificada; como resultado, todas as mensagens enviadas para o cliente do jogo usam TCP.
- **CONNECTED_SEND_AND_RECEIVE_FAST** — O cliente do jogo está conectado ao servidor com uma conexão TCP e uma conexão UDP. Realtime O cliente do jogo pode enviar e receber mensagens usando TCP ou UDP.
- **CONNECTING** O cliente do jogo enviou uma solicitação de conexão e o servidor Realtime a está processando.
- **DISCONNECTED_CLIENT_CALL** — O cliente do jogo foi desconectado do Realtime servidor em resposta a uma solicitação do cliente do jogo. [Disconnect\(\)](#)
- **DESCONECTADO** — O cliente do jogo foi desconectado do Realtime servidor por um motivo diferente de uma chamada de desconexão do cliente.

ConnectionType

- **TLS12_RT_OVER_WSS_DTLS_** — Tipo de conexão segura.

Para uso com servidores em tempo real que estão sendo executados em uma Amazon GameLift Servers frota com um certificado TLS gerado. Ao usar uma conexão segura, o tráfego TCP é criptografado usando TLS 1.2 e o tráfego UDP é criptografado usando DTLS 1.2.

- **RT_OVER_WS_UDP_UNSECURED**: tipo de conexão não segura.
- **RT_OVER_WEBSOCKET**: tipo de conexão não segura. Esse valor não é mais preferível.

DeliveryIntent

- **FAST**: entregue usando um canal UDP.
- **RELIABLE**: entregue usando uma conexão TCP.

Referência de script para Amazon GameLift ServersRealtime

Use esses recursos para criar lógica personalizada em seus scripts Realtime.

Tópicos

- [Retornos de chamada de script para Amazon GameLift ServersRealtime](#)
- [Amazon GameLift ServersRealtimeinterface](#)

Retornos de chamada de script para Amazon GameLift ServersRealtime

Você pode fornecer uma lógica personalizada para responder a eventos implementando esses retornos de chamada em seu script do Realtime.

Init

Inicializa o servidor Realtime e recebe uma interface de servidor Realtime.

Sintaxe

```
init(rtsession)
```

onMessage

Invocado quando uma mensagem recebida é enviada para o servidor.

Sintaxe

```
onMessage(gameMessage)
```

onHealthCheck

Invocada para definir o status de integridade da sessão de jogo. Por padrão, o status de integridade é íntegro (ou `true`). Esse retorno de chamada pode ser implementado para executar verificações de integridade personalizadas e retornar um status.

Sintaxe

```
onHealthCheck()
```

onStartGameSessão

Invocada quando uma nova sessão de jogo é iniciada, com um objeto de sessão do jogo transmitido.

Sintaxe

```
onStartGameSession(session)
```

onProcessTerminate

Chamada quando o processo do servidor está sendo encerrado pelo serviço Amazon GameLift Servers. Pode atuar como um trigger para sair corretamente da sessão do jogo. Não há necessidade de chamar `processEnding()`.

Sintaxe

```
onProcessTerminate()
```

onPlayerConnect

Invocado quando um jogador solicita uma conexão e passou a validação inicial.

Sintaxe

```
onPlayerConnect(connectMessage)
```

onPlayerAccepted

Invocado quando uma conexão do jogador é aceita.

Sintaxe

```
onPlayerAccepted(player)
```

onPlayerDisconnect

Invocado quando um jogador se desconecta da sessão do jogo, enviando uma solicitação de desconexão ou por outros meios.

Sintaxe

```
onPlayerDisconnect(peerId)
```

onProcessStarted

Invocado quando um processo do servidor é iniciado. Esse retorno de chamada permite que o script personalizado execute as tarefas necessárias para se preparar para hospedar uma sessão de jogo.

Sintaxe

```
onProcessStarted(args)
```

onSendToJogador

Invocado quando uma mensagem é recebida no servidor de um jogador para ser entregue para outro jogador. Esse processo é executado antes que a mensagem seja entregue.

Sintaxe

```
onSendToPlayer(gameMessage)
```

onSendToGrupo

Invocado quando uma mensagem é recebida no servidor de um jogador para ser entregue para um grupo. Esse processo é executado antes que a mensagem seja entregue.

Sintaxe

```
onSendToGroup(gameMessage))
```

onPlayerJoinGrupo

Invocado quando um jogador envia uma solicitação para ingressar em um grupo.

Sintaxe

```
onPlayerJoinGroup(groupId, peerId)
```

onPlayerLeaveGrupo

Invocado quando um jogador envia uma solicitação para sair de um grupo.

Sintaxe

```
onPlayerLeaveGroup(groupId, peerId)
```

Amazon GameLift ServersRealtimeinterface

Quando um Amazon GameLift Servers Realtime script é inicializado, uma interface para o Realtime servidor é retornada. Este tópico descreve as propriedades e métodos disponíveis por meio da

interface. Saiba mais sobre como escrever scripts Realtime e visualize um script detalhado de exemplo no [Criação de um Realtime script](#).

A interface Realtime fornece acesso aos seguintes objetos:

- sessão
- player
- gameMessage
- configuration

Objeto de sessão Realtime

Use esses métodos para acessar informações relacionadas ao servidor e para executar ações relacionadas ao servidor.

getPlayers()

Recupera uma lista de pares IDs para jogadores que estão atualmente conectados à sessão do jogo. Retorna uma matriz de objetos de jogador.

Sintaxe

```
rtSession.getPlayers()
```

broadcastGroupMembershipAtualizar ()

Aciona a entrega de uma lista atualizada de associações do grupo ao grupo de jogadores. Especifique qual associação groupIdTo transmitir (Transmitir) e o grupo para receber a atualização (targetGroupId). O grupo IDs deve ser um número inteiro positivo ou "-1" para indicar todos os grupos. Veja um [Amazon GameLift ServersRealtimeexemplo de script](#) exemplo de grupo IDs definido pelo usuário.

Sintaxe

```
rtSession.broadcastGroupMembershipUpdate(groupIdToBroadcast, targetGroupId)
```

getServerId()

Recupera o identificador do ID par exclusivo do servidor, que é usado para rotear mensagens para o servidor.

Sintaxe

```
rtSession.getServerId()
```

getAllPlayersGroupId()

Recupera o ID de grupo para o grupo padrão que contém todos os jogadores conectados no momento à sessão de jogo.

Sintaxe

```
rtSession.getAllPlayersGroupId()
```

processEnding()

Aciona o servidor Realtime para encerrar o servidor de jogos. Essa função deve ser chamada no script Realtime para sair corretamente de uma sessão de jogo.

Sintaxe

```
rtSession.processEnding()
```

getGameSessionIdentificação ()

Recupera o ID exclusivo da sessão do jogo em execução no momento.

Sintaxe

```
rtSession.getGameSessionId()
```

getLogger()

Recupera a interface para o registro em log. Use essa opção para registrar instruções de log que serão capturadas em seus logs de sessão de jogo. O registrador em log é compatível com o uso de instruções “info”, “warn” e “error”. Por exemplo: `logger.info("<string>")`.

Sintaxe

```
rtSession.getLogger()
```

sendMessage()

Envia uma mensagem, criada usando `newTextGameMessage` ou `newBinaryGameMessage`, do servidor Realtime para um jogador destinatário usando o canal UDP. Identifique o destinatário usando o ID par do jogador.

Sintaxe

```
rtSession.sendMessage(gameMessage, targetPlayer)
```

sendGroupMessage()

Envia uma mensagem, criada usando `newTextGameMessage` ou `newBinaryGameMessage`, do servidor Realtime a todos os jogadores em um grupo de jogadores usando o canal UDP. O grupo IDs deve ser um número inteiro positivo ou "-1" para indicar todos os grupos. Veja um [Amazon GameLift ServersRealtimeexemplo de script](#) exemplo de grupo IDs definido pelo usuário.

Sintaxe

```
rtSession.sendGroupMessage(gameMessage, targetGroup)
```

sendReliableMessage()

Envia uma mensagem, criada usando `newTextGameMessage` ou `newBinaryGameMessage`, do servidor Realtime para um jogador destinatário usando o canal TCP. Identifique o destinatário usando o ID par do jogador.

Sintaxe

```
rtSession.sendReliableMessage(gameMessage, targetPlayer)
```

sendReliableGroupMensagem ()

Envia uma mensagem, criada usando `newTextGameMessage` ou `newBinaryGameMessage`, do servidor Realtime para todos os jogadores em um grupo de jogadores usando o canal TCP. Grupo IDs que deve ser um número inteiro positivo ou "-1" para indicar todos os grupos. Veja um [Amazon GameLift ServersRealtimeexemplo de script](#) exemplo de grupo IDs definido pelo usuário.

Sintaxe

```
rtSession.sendReliableGroupMessage(gameMessage, targetGroup)
```

newTextGameMensagem ()

Cria uma nova mensagem contendo texto, a ser enviada do servidor para os destinatários do jogador usando as SendMessage funções. O formato da mensagem é semelhante ao formato usado no SDK do cliente para Realtime (consulte [RTMessage](#)). Informa um objeto gameMessage.

Sintaxe

```
rtSession.newTextGameMessage(opcode, sender, payload)
```

newBinaryGameMensagem ()

Cria uma nova mensagem contendo dados binários, a serem enviados do servidor para os destinatários do jogador usando as SendMessage funções. O formato da mensagem é semelhante ao formato usado no SDK do cliente para Realtime (consulte [RTMessage](#)). Informa um objeto gameMessage.

Sintaxe

```
rtSession.newBinaryGameMessage(opcode, sender, binaryPayload)
```

Objeto do jogador

Acessar informações relacionadas ao jogador.

player.peerId

ID exclusivo que é atribuído a um cliente do jogo quando ele se conecta ao servidor Realtime e ingressa na sessão do jogo.

jogador. playerId

O ID da sessão do jogador que foi referenciado pelo cliente do jogo quando conectou-se ao servidor Realtime e ingressou na sessão do jogo.

Objeto de mensagem de jogo

Use esses métodos para acessar as mensagens que são recebidas pelo servidor Realtime. As mensagens recebidas de clientes de jogos têm a estrutura [RTMessage](#).

getPayloadAsTexto ()

Obtém a carga útil da mensagem do jogo como texto.

Sintaxe

```
gameMessage.getPayloadAsText()
```

`gameMessage.opcode`

Código de operação contido em uma mensagem.

`gameMessage.payload`

Carga contida em uma mensagem. Pode ser texto ou binário.

`gameMessage.sender`

ID par do cliente do jogo que enviou uma mensagem.

`gameMessage.reliable`

Booleano indica se a mensagem foi enviada por meio de TCP (verdadeiro) ou UDP (falso).

Objeto de configuração

O objeto de configuração pode ser usado para substituir as configurações padrão.

`configuration.maxPlayers`

O número máximo de conexões cliente/servidor que podem ser aceitas pelo RealTimeServers.

O padrão é 32.

`configuração.pingIntervalTime`

Intervalo de tempo em milissegundos em que o servidor tentará enviar um ping a todos os clientes conectados para verificar se as conexões estão íntegras.

O padrão é 3000 ms.

Eventos de posicionamento de sessões de jogo

Amazon GameLift Servers emite eventos para cada solicitação de colocação de sessão de jogo à medida que ela é processada. Você pode publicar esses eventos em um tópico do Amazon SNS, conforme descrito em [Configurar notificações de eventos para o posicionamento de sessões de jogo](#).

Esses eventos também são transmitidos para a Amazon CloudWatch Events quase em tempo real e com base no melhor esforço.

Este tópico descreve a estrutura dos eventos de posicionamento de sessões de jogo e fornece um exemplo para cada tipo de evento. Para obter mais informações sobre o status das solicitações de colocação de sessões de jogo, consulte [GameSessionPlacement](#) no Amazon GameLift Servers Referência da API.

Sintaxe do evento de posicionamento

Os eventos são representados como objetos JSON. A estrutura do evento está em conformidade com o padrão CloudWatch Eventos, com campos de nível superior semelhantes e detalhes específicos do serviço.

Os campos de nível superior incluem o seguinte (consulte o [padrão do evento](#) para obter mais detalhes):

version

Esse campo é sempre definido como 0 (zero).

id

Identificador de rastreamento exclusivo do evento.

detail-type

O valor é sempre `GameLift Queue Placement Event`.

origem

O valor é sempre `aws.gamelift`.

conta

A AWS conta que está sendo usada para gerenciar Amazon GameLift Servers.

horário

Carimbo de data/hora do evento.

região

A AWS região em que a solicitação de colocação está sendo processada. Essa é a região em que a fila de sessões de jogos em uso reside.

resources

Valor ARN da fila da sessão de jogo que está processando a solicitação de posicionamento.

PlacementFulfilled

A solicitação de posicionamento foi atendida com sucesso. Uma nova sessão de jogo foi iniciada e novas sessões de jogadores foram criadas para cada jogador listado na solicitação de posicionamento de sessões de jogo. As informações de conexão do jogador estão disponíveis.

Sintaxe detalhada:

placementId

Identificador exclusivo atribuído à solicitação de posicionamento de sessões de jogo.

porta

Número da porta da nova sessão de jogos.

gameSessionArn

Identificador ARN da nova sessão de jogos.

ipAddress

Endereço IP da sessão de jogos.

dnsName

Identificador DNS atribuído à instância que está executando a nova sessão de jogos. O formato do valor é diferente dependendo se a instância que executa a sessão de jogos está habilitada para TLS. Ao se conectar a uma sessão de jogos em uma frota habilitada para TLS, os jogadores devem usar o nome DNS e não o endereço IP.

Frotas habilitadas para TLS: <unique identifier>.<region identifier>.amazongamelift.com.

Non-TLS-enabled frotas: ec2-<unique identifier>.compute.amazonaws.com.

startTime

Carimbo de data/hora indicando quando essa solicitação foi posicionada na fila.

endTime

Carimbo de data/hora indicando quando essa solicitação foi atendida.

gameSessionRegion

AWS Região da frota que está hospedando a sessão do jogo. Isso corresponde ao token de região no GameSessionArn.

placedPlayerSessions

Coleção de sessões de jogadores que foram criadas para cada jogador na solicitação de posicionamento de sessões de jogo.

Exemplo

```
{
  "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/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",
    "placedPlayerSessions": [
      {
        "playerId": "player-1"
        "playerSessionId": "psess-1232131232324124123123"
      }
    ]
  }
}
```

```
]
}
}
```

PlacementCancelled

A solicitação de colocação foi cancelada com uma ligação para o GameLift serviço [StopGameSessionPlacement](#).

Detalhe:

placementId

Identificador exclusivo atribuído à solicitação de posicionamento de sessões de jogo.

startTime

Carimbo de data/hora indicando quando essa solicitação foi posicionada na fila.

endTime

Carimbo de data/hora indicando quando essa solicitação foi cancelada.

Exemplo

```
{
  "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

O posicionamento de sessões de jogo não foi concluído com êxito antes que o limite de tempo da fila expirasse. A solicitação de posicionamento pode ser reenviada conforme necessário.

Detalhe:

placementId

Identificador exclusivo atribuído à solicitação de posicionamento de sessões de jogo.

startTime

Carimbo de data/hora indicando quando essa solicitação foi posicionada na fila.

endTime

Carimbo de data/hora indicando quando essa solicitação foi cancelada.

Exemplo

```
{  
  "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 não foi possível atender à solicitação da sessão do jogo. Isso geralmente é causado por um erro interno inesperado. A solicitação de posicionamento pode ser reenviada conforme necessário.

Detalhe:

placementId

Identificador exclusivo atribuído à solicitação de posicionamento de sessões de jogo.

startTime

Carimbo de data/hora indicando quando essa solicitação foi posicionada na fila.

endTime

Carimbo de data/hora indicando quando essa solicitação falhou.

Exemplo

```
{
  "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 Servers Versões da AMI

A tabela a seguir identifica as imagens de máquina mais recentes da Amazon (AMIs) Amazon GameLift Servers usadas para EC2 hospedagem gerenciada. Conforme descrito em [Análise de configuração e vulnerabilidade em Amazon GameLift Servers](#), você deve criar regularmente novas EC2 frotas Amazon GameLift Servers gerenciadas para implantar as atualizações mais recentes da versão da AMI.

AMIs para uso com o SDK do servidor para Amazon GameLift Servers maiores de 5 anos

Amazon GameLift Servers usa o seguinte AMIs para hospedar servidores de jogos integrados ao SDK do servidor para a Amazon GameLift Servers versão 5.

Imagem da Amazon GameLift Servers	Arquitetura	Patch mais recente	AWS Imagem base
Amazon Linux 2023 BASE_AMI_LINUX_2023	x86	2025-04-22	Amazon Linux 2023 AMI 2023.6.20 250317.2 x86_64 HVM kernel-6.1 (notas de lançamento)
Amazon Linux 2 BASE_AMI_LINUX_2_SDK_5	x86	2025-04-22	Amazon Linux 2 Kernel 5.10 AMI 2.0.20250321.0 x86_64 HVM gp2 (notas de lançamento)
Windows 2016 BASE_AMI_WINDOWS_2016_SDK_5	x86	2025-04-22	Windows_Server-2016-English-Full-Base-2025.03.12 (notas de lançamento)
Amazon Linux 2023 BASE_AMI_LINUX_2023_ARM	ARM64	2025-04-22	Amazon Linux 2023 AMI 2023.6.20 250317.2 arm64 HVM kernel-6.1 (notas de lançamento)
Amazon Linux 2 BASE_AMI_LINUX_2_ARM	ARM64	2025-04-22	Amazon Linux 2 LTS Arm64 Kernel 5.10 AMI 2.0.20250321.0 arm64 HVM gp2 (notas de lançamento)

AMIs para uso com o SDK do servidor para 4 Amazon GameLift Servers

Amazon GameLift Servers usa o seguinte AMIs para hospedar servidores de jogos integrados ao SDK do servidor para a Amazon GameLift Servers versão 4 ou anterior.

Imagem da Amazon GameLift Servers	Arquitetura	Patch mais recente	AWS Imagem base
Amazon Linux 2 BASE_AMI_LINUX_2_SDK_4	x86	2025-04-22	Amazon Linux 2 AMI 2.0.20250321.0 x86_64 HVM gp2 (notas de lançamento)
Windows 2016 BASE_AMI_WINDOWS_2016_SDK_4	x86	2025-04-22	Windows_Server-2016-English-Full-Base-2025.03.12 (notas de lançamento)

Para obter mais informações, consulte os seguintes recursos:

- [Notas de lançamento do Amazon Linux 2023](#)
- [Notas de lançamento do Amazon Linux 2](#)
- [AWS Histórico de versões da AMI do Windows](#)
- [Descrição das alterações do conteúdo de serviços de atualização de software e do Windows Server para 2024](#)

Endpoints e cotas do Amazon GameLift Servers

- [Para pontos finais de serviço e pontos de extremidade de beacon que você pode usar para se conectar programaticamente ao serviço, consulte pontos de extremidade de serviço. Amazon GameLift Servers](#)
- Para Amazon GameLift Servers cotas sobre o uso de recursos de serviço ou operações por AWS conta, consulte [cotas Amazon GameLift Servers de serviço](#). Para obter mais informações sobre as cotas e como solicitar um aumento, consulte [Service Quotas da AWS](#). Você pode solicitar aumentos de cota usando o Amazon GameLift Servers console.

Amazon GameLift ServersSinalizadores de ping UDP

Os beacons de ping UDP fornecem uma maneira de medir a latência da rede entre os dispositivos dos jogadores e os locais de hospedagem. Amazon GameLift Servers Com sinalizadores de ping, você pode coletar dados de latência precisos para tomar decisões informadas sobre o posicionamento do servidor do jogo e melhorar a combinação de jogadores com base nos requisitos de latência.

Como funcionam os sinalizadores de ping UDP

Amazon GameLift Servers fornece endpoints UDP fixos (beacons de ping) em cada local de hospedagem onde você pode implantar servidores de jogos. Como a maioria dos servidores de jogos se comunica usando UDP, medir a latência com beacons de ping UDP fornece resultados mais precisos do que usando pings ICMP. Os dispositivos de rede geralmente lidam com pacotes ICMP de forma diferente dos pacotes UDP, o que pode levar a medições de latência que não refletem o verdadeiro desempenho que seus jogadores experimentarão.

Com os beacons de ping UDP, seu cliente de jogo pode enviar mensagens UDP para esses endpoints e receber respostas assíncronas, fornecendo medidas de latência que representam melhor as condições reais de tráfego do jogo entre o dispositivo de um jogador e possíveis locais de hospedagem. Os endpoints são permanentes e permanecem disponíveis desde que ofereçam Amazon GameLift Servers suporte à hospedagem de jogos nesse local.

Casos de uso comuns para beacons de ping UDP

Você pode usar beacons de ping UDP de várias maneiras para otimizar a experiência de rede do seu jogo.

Escolhendo locais de hospedagem ideais

Colete dados de latência em diferentes regiões geográficas para identificar os melhores locais primários e de backup para hospedar servidores de jogos para sua base de jogadores.

Colocar sessões de jogo com base na latência do jogador

Inclua dados de latência do jogador ao solicitar novas sessões de jogo para ajudar a escolher locais que ofereçam a experiência de menor latência.

Otimizando o matchmaking com base na latência

Forneça dados de latência do jogador ao solicitar matchmaking para ajudar a combinar jogadores com perfis de latência semelhantes e colocar as sessões de jogo em locais ideais para jogadores pareados.

Note

Ao criar solicitações de matchmaking, você não deve fornecer informações de latência para locais onde você não tem frotas. Se você fizer isso, Amazon GameLift Servers pode tentar colocar a sessão do jogo em locais onde não há capacidade da frota, resultando em falhas na solicitação de matchmaking.

Obtendo endpoints de beacon

Para recuperar as informações do domínio e da porta do sinalizador de ping para Amazon GameLift Servers locais, use a operação da [ListLocations](#) API. O conjunto de locais retornados por essa API depende do Região da AWS que você especifica ao chamá-la (ou da sua região padrão, se você não especificar uma). Quando você liga de:

- Uma região de origem de uma frota que suporta vários locais: a API retorna informações de todos os locais de hospedagem
- Uma região de origem de uma frota que suporta um único local: a API retorna informações desse local

Observe que, se você chamar essa API usando um local que só pode ser remoto em uma frota com vários locais, a API retornará um erro porque esse tipo de local não tem um ponto final de serviço.

Consulte a tabela de locais compatíveis [AWS Localizações suportadas](#) para identificar as regiões de origem que oferecem suporte a frotas de um ou vários locais.

Exemplo

```
aws gamelift list-locations --region ap-northeast-2
```

Isso Região da AWS suporta frotas com vários locais, portanto, vários locais serão devolvidos. Aqui está um exemplo de um dos valores de retorno:

```
[...]
```



```
{
  "LocationName": "ap-northeast-1",
  "PingBeacon": {
    "UDPEndpoint": {
      "Domain": "gamelift-ping.ap-northeast-1.api.aws",
      "Port": 7770
    }
  }
}
```

Important

Armazene em cache as informações do sinalizador de ping em vez de ligar `ListLocations` antes de cada medição de latência. As informações do domínio e da porta são estáticas e a API não foi projetada para solicitações de alto volume.

Implementando medições de latência

Siga estas melhores práticas ao implementar medições de latência usando beacons de ping UDP:

1. Armazene as informações do sinalizador de ping usando uma das seguintes abordagens:

- Codifique os endpoints em seu cliente de jogo.
- Armazene as informações no back-end do seu jogo.
- Implemente um mecanismo de atualização periódica (diária/semanal) para atualizar as informações.

2. Envie mensagens de ping UDP:

- Coloque o que quiser no corpo da mensagem, desde que não esteja vazio, e você manterá as mensagens abaixo do tamanho máximo de 300 bytes.
- Observe os seguintes limites de tarifa para cada local:
 - 3 transações por segundo (TPS) por combinação exclusiva de endereço IP e porta do remetente
 - 1000 TPS por endereço IP exclusivo do remetente

3. Calcule a latência:

- Envie vários pings para cada local para calcular uma latência média.
- Considere enviar pings simultâneos para vários locais para obter resultados mais rápidos.

- Use a lógica de repetição conforme necessário para enviar novos pacotes para qualquer pacote que não tenha sido devolvido em pouco tempo (normalmente de 1 a 3 segundos), pois o UDP não tem entrega 100% garantida.
- Calcule a diferença de tempo entre enviar uma mensagem e receber a resposta.
- Se uma grande parte dos pings UDP para um local consistentemente não obtiverem uma resposta, calcule a latência usando pings ICMP para nossos terminais de [Amazon GameLift Servers](#) padrão como alternativa.

Tip

Recomendamos que você inclua a porta 7770 sempre que documentar a lista de portas que seus players devem ter abertas na rede local. Esse é outro motivo pelo qual você deve ter um substituto para medir a latência (usando ICMP, por exemplo) caso essa porta esteja bloqueada.

Exemplos de código

Aqui estão alguns exemplos simples que mostram como enviar pings UDP e calcular a latência.

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($"\\nSummary:");
        Console.WriteLine($"Successful pings: {successfulPings}/
{numPings}");
        Console.WriteLine($"Average latency: {averageLatency:F2}ms");
    }
    else
    {
        Console.WriteLine("\\nNo successful pings");
    }
}
catch (Exception ex)
{
    Console.WriteLine($"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()
```

Notas de release do Amazon GameLift Servers

As notas de release do Amazon GameLift Servers fornecem detalhes sobre novos recursos, atualizações e correções relacionadas ao serviço.

Versões do SDK

As tabelas a seguir listam todas as Amazon GameLift Servers versões com informações sobre a versão do SDK. Não há necessidade de uso comparável SDKs para suas integrações de servidor de jogos e cliente. No entanto, as versões anteriores de um SDK podem não oferecer suporte total aos recursos mais recentes em outro.

Amazon GameLift Servers SDKs e os plug-ins são de código aberto. Consulte [Obtenha ferramentas Amazon GameLift Servers de desenvolvimento](#). Para obter as versões mais recentes, consulte a [Amazon GameLift Servers GitHub organização](#).

Versão atual

Requisitos
SDKs
Unidades
cliente
Plataforma
(C#)
+
+))
52
ou
posterior

Versões anteriores

[illegible]

Amazon GameS

Amazon
United
cliente

Idade
3#
+
Irreal

posterior

[5210151](#)

ou
posterior

[5210251](#)

ou
posterior

[5210251](#)

ou
posterior

[5210251](#)

ou
posterior

[521031](#)

ou
posterior

[510030](#)

ou
posterior

[510042](#)

ou
posterior

Introdução

elbirtéyes

cliente

Cidade

Ⓢ#

+

Irreal

~~SECRET~~ B41

bu

posterior

510000712

ou

posterior

510340642

U

posterior

~~510340~~6-1

ou

posterior

~~510320~~ 5-2

U

posterior

~~513200~~ 513200-2

ou

posterior

~~50C200~~ 50C204-1

U

posterior

[801200-0](#)
Unreal Engine

[801200-0](#)
Unreal Engine
cliente

[801200-0](#)
Unreal Engine
3#
+
Unreal

[801200-0](#)
Unreal Engine
ou
posterior

[801200-0](#)
Unreal Engine
ou
posterior

[801200-0](#)
Unreal Engine
NET
Hosted
)

[801200-0](#)
Unreal Engine
ou
posterior

[801200-0](#)
Unreal Engine
ou
posterior

[801200-0](#)
Unreal Engine
ou
posterior

[801200-0](#)
Unreal Engine
ou
posterior

AmazonS3

Amazon
Unity
cliente

Unidade
3#
+
Irreal

[821303-1](#)
ou
posterior

[821302-0](#)
ou
posterior

[821302-2](#)
ou
posterior

[821201-2](#)
ou
posterior

[821201-1](#)
ou
posterior

[821200-1](#)
ou
posterior

[821100-2](#)
ou
posterior

Introduction

előmozdítás

cliente

Cidade

Ⓢ#

+

Irreal

32300-1

you

posterior

3.41.000

libu

posterior

3.41001

!ou

posterior

B.71.000 -1

U

posterior

3.71000-2

you

posterior

3.74.000-0

ibu

posterior

3300-0

!ou

posterior

AmazonS3

Amazon
Unity
cliente

Unidade
3#
+
Irreal

[B30001](#)
ou
posterior

[B3003-0](#)
ou
posterior

[B3002-0](#)
ou
posterior

[B300201](#)
ou
posterior

[B2209-2](#)
ou
posterior

[B2206-1](#)
ou
posterior

[B2205-1](#)
ou
posterior

Introdução

elbirtéyes

cliente

Cidade

Ⓢ#

+

Irreal

3.2.11

ou

posterior

3.2.0

U

posterior

B.11.7709-0

4

posterior

B.11.87/201

u

posterior

~~B.01.7525-1~~

4

posterior

3.017504-1

44

posterior

3.11.5

ou

posterior

Sistema

NomeS

Cidade

#

+ Irreal

301691-1-1

libu posterior

301697-10-1

libu posterior

31082201

libu posterior

310871201

libu posterior

Notas da versão

As notas de release a seguir estão em ordem cronológica, com as atualizações mais recentes listadas primeiro. O Amazon GameLift Servers foi lançado pela primeira vez em 2016. Para obter notas de release anteriores às listadas aqui, consulte os links de datas de lançamento em [Versões do SDK](#).

24 de junho de 2025: Amazon GameLift Servers lança beacons de ping UDP para ajudar a medir a latência da rede em tempo real para jogos multijogador

Versões atualizadas do SDK:

- SDK DA AWS 1.11.595

Amazon GameLift Servers libera para disponibilidade geral um conjunto de endpoints fixos chamados beacons de ping UDP para ajudá-lo a medir com precisão a latência entre os dispositivos dos jogadores e as localizações dos servidores do jogo. Os endpoints do sinalizador de ping UDP estão disponíveis em todas as Regiões AWS Globais e Zonas Locais suportadas pela Amazon GameLift Servers, com exceção das Regiões da China. AWS

A maioria dos jogos multijogador usa UDP (User Datagram Protocol) como seu principal protocolo de transmissão de pacotes devido aos benefícios de desempenho para jogos em tempo real. Compreender e otimizar a latência da rede é crucial para oferecer a melhor experiência possível ao jogador. Os beacons de ping UDP fornecem uma maneira consistente e confiável de medir a latência real do pacote UDP entre jogadores e servidores de jogos, ajudando você a tomar melhores decisões sobre player-to-server a combinação e o posicionamento da sessão do jogo.

A [ListLocations](#) API foi estendida para incluir informações de domínio e porta do endpoint como parte da lista de locais que ela retorna, facilitando o acesso programático aos endpoints.

Seu cliente de jogo pode enviar mensagens UDP para esses endpoints e receber respostas assíncronas com os mesmos dados, fornecendo medidas de latência que representam melhor as condições reais de tráfego do jogo entre o dispositivo de um jogador e possíveis locais de hospedagem. Esses endpoints são permanentes e permanecem disponíveis desde que ofereçam Amazon GameLift Servers suporte à hospedagem de jogos nesse local.

Saiba mais:

- [Sinalizadores de ping UDP](#), Guia Amazon GameLift Servers do desenvolvedor
- [ListLocations](#), Referência Amazon GameLift Servers da API

24 de junho de 2025: Amazon GameLift Servers adiciona suporte de localização para a Tailândia e a Malásia

Com a hospedagem Amazon GameLift Servers gerenciada, agora você pode implantar recursos de servidores de jogos em Bangkok, Tailândia e Kuala Lumpur, Malásia, estendendo o alcance de seus jogos a jogadores de todo o Sudeste Asiático. Essas novas regiões ajudam a reduzir a latência e melhorar a experiência de jogo dos jogadores nessas áreas.

Os itens a seguir Regiões da AWS estão disponíveis como locais remotos para frotas com vários locais. Para começar a hospedar sessões de jogo nesses locais, adicione-os como locais remotos a uma frota nova ou existente em vários locais. Com frotas de vários locais, você pode gerenciar diretamente a capacidade de hospedagem em cada local.

- [Ásia-Pacífico \(Tailândia\) ap-southeast-7](#) ()
- [Ásia-Pacífico \(Malásia\) ap-southeast-5](#) ()

Eles não Regiões da AWS estão habilitados por padrão para uma AWS conta. Você deve se cadastrar em cada região antes de poder implantar Amazon GameLift Servers recursos lá.

Saiba mais:

- [Amazon GameLift Servers locais de serviço](#), Guia do desenvolvedor de Amazon GameLift Servers hospedagem
- [Atualizar localizações da frota](#), Guia do desenvolvedor de Amazon GameLift Servers hospedagem
- [Ative ou desative AWS regiões em sua AWS conta](#), Guia de referência de gerenciamento de contas.

29 de maio de 2025: Amazon GameLift Servers lança novas versões do SDK do servidor e plug-ins para Unreal Engine e Unity

Versões atualizadas do SDK:

- [SDK do servidor C++, versão 5.3.0](#)
- [SDK do servidor C#, versão 5.3.0](#)
- [SDK do servidor Go, versão 5.3.0](#)
- [SDK do servidor para Unreal, versão 5.3.0](#)
- [SDK de servidor para Unity, versão 5.3.0](#)

Versões atualizadas do plug-in:

- [Plugin para Unreal](#), versão 3.0.0
- [Plugin para Unity](#), versão 3.1.0

Novas versões do SDK do servidor para C++, C#, Go, Unreal e Unity, bem como novas versões de plug-ins para Unreal Engine e Unity, agora são de código aberto. Todos eles estão disponíveis na [Amazon GameLift Servers GitHub organização](#). Para obter informações detalhadas sobre a atualização, consulte as notas de versão e os readmes em cada repositório.

Principais atualizações do SDK do servidor:

- Validação aprimorada do lado do cliente e respostas de erro em todos os servidores. SDKs
- A função de [OnProcessTerminate](#) retorno de chamada agora tem lógica padrão para finalizar o processo do servidor do jogo.
- A função [InitSDK\(\)](#) agora usa um token de idempotência para oferecer suporte a várias tentativas.
- O [OnUpdateGameSession](#) retorno de chamada agora pode transmitir dados para `autoBackfillMode`.

Principais atualizações do plugin:

- O plug-in do Unreal Engine agora tem um processo de instalação e configuração mais simplificado, com mais automação e menos pré-requisitos (CMake, OpenSSL e a cadeia de ferramentas de compilação cruzada Unreal).
- O plug-in do Unreal Engine aprimorou a experiência de interface do usuário para o EC2 fluxo de trabalho gerenciado, incluindo suporte para espaços nos caminhos de criação do cliente e do servidor do jogo. Além disso, agora você pode adicionar argumentos de linha de comando ao iniciar um cliente de jogo a partir do Editor.
- O plug-in do Unreal Engine agora oferece suporte a compilações de servidores ARM. UE5

Saiba mais:

- [Amazon GameLift Serversplugin para Unreal Engine](#), Guia Amazon GameLift Servers do desenvolvedor
- [Amazon GameLift Serversplugin para Unity \(servidor SDK 5.x\)](#), Guia Amazon GameLift Servers do desenvolvedor
- [SDK de servidor 5.x para Amazon GameLift Servers](#), Guia Amazon GameLift Servers do desenvolvedor
- [Amazon GameLift ServersIntegre-se a um projeto Unreal Engine](#), Guia Amazon GameLift Servers do desenvolvedor

- [Amazon GameLift ServersIntegre-se a um projeto Unity](#), Guia Amazon GameLift Servers do desenvolvedor

24 de abril de 2025: o SDK 5 do servidor C#, por Amazon GameLift Servers enquanto, oferece suporte a o.NET 8

Versões atualizadas do SDK:

- SDK do servidor C#, versão 5.2.1

Para desenvolvedores de jogos que usam C#, agora você pode usar o.NET 8 como uma estrutura de destino para seus Amazon GameLift Servers projetos. Com o.NET 8, você pode aproveitar as melhorias de desempenho, incluindo compilação aprimorada just-in-time (JIT), otimização do uso de memória e tempos de inicialização mais rápidos. Se você usa atualmente o.NET 6, recomendamos que planeje uma migração para o.NET 8, incluindo a atualização do SDK do servidor C# para a versão mais recente. A Microsoft anunciou o suporte ao .NET 8, com patches de segurança e atualizações técnicas, até novembro de 2026.

[Baixe a versão mais recente do SDK do servidor C# para Amazon GameLift Servers](#)

Saiba mais:

- [Obtenha ferramentas Amazon GameLift Servers de desenvolvimento](#), Guia Amazon GameLift Servers do desenvolvedor

27 de março de 2025: Amazon GameLift Servers expande o suporte para famílias de instâncias de próxima geração EC2

Versões atualizadas do SDK:

- AWS SDK 1.11.535

Agora você pode ajustar a hospedagem do seu servidor de jogos Amazon GameLift Servers selecionando entre uma seleção maior de instâncias da Amazon nas famílias de EC2 instâncias da 5ª à 8ª geração. Cada nova EC2 geração oferece avanços em EC2 computação, memória e rede, com as instâncias da 8ª geração oferecendo instâncias de ponta AWS baseadas em Graviton4 e Intel Xeon. As instâncias de próxima geração estão disponíveis nas seguintes famílias de instâncias:

- [Uso geral](#) (série M)
- [Otimizado para computação](#) (série C)
- [Memória otimizada](#) (série R)

Você também pode escolher variantes que ofereçam armazenamento local (d), rede aprimorada (n) e arquiteturas de processador específicas (Intel/AMD/Graviton – i/a/g). As instâncias de próxima geração estão disponíveis em todas as Regiões da AWS que são suportadas pela Amazon GameLift Servers, com exceção das regiões da AWS China. Consulte mais detalhes em [Amazon GameLift Servers locais de serviço](#).

Use esses novos tipos de instância com EC2 frotas Amazon GameLift Servers gerenciadas e frotas de contêineres gerenciadas. Ao mudar sua hospedagem de jogos existente para um novo tipo de instância (mesma arquitetura), basta implantar novas frotas com todas as configurações inalteradas, exceto o tipo de instância.

Saiba mais:

- [Amazon GameLift Servers Preços de instâncias](#)
- [CreateFleet](#) Amazon GameLift Servers Referência da API
- [CreateContainerFleet](#), Referência Amazon GameLift Servers da API

11 de março de 2025: Amazon GameLift Servers lança o novo Game Server Wrapper para integração rápida

O novo invólucro de servidor de jogos reduz Amazon GameLift Servers significativamente o tempo necessário para hospedar um servidor de jogos. Amazon GameLift Servers Sem a necessidade de alterações no código, você pode usar o wrapper para adicionar a funcionalidade básica de gerenciamento de sessões de jogo ao seu jogo e implantá-la Amazon GameLift Servers em uma frota Anywhere, frota gerenciada ou EC2 frota de contêineres gerenciada. Essa ferramenta é ideal para fazer uma avaliação prática dos Amazon GameLift Servers recursos, usando seu próprio servidor de jogo ou um de um jogo de amostra. Também é útil para implantar rapidamente iterações de servidores de jogos, como para prototipagem ou testes rápidos.

Com os recursos básicos de gerenciamento de sessão de jogo, seu servidor de jogo pode inicializar uma conexão com o Amazon GameLift Servers serviço, responder às solicitações para iniciar e interromper as sessões de jogo e desligar quando uma sessão de jogo for concluída.

[Baixe o invólucro do servidor de GitHub jogos em.](#) Consulte o readme GitHub para saber como instalar e usar o wrapper com seus projetos de jogo.

6 de março de 2025: Amazon GameLift é agora Amazon GameLift Servers e Amazon GameLift Streams

GameLift A Amazon é um serviço totalmente gerenciado que se dedica a ajudar os desenvolvedores a criar, escalar e entregar os jogos mais exigentes do mundo. Com o lançamento de disponibilidade geral do Amazon GameLift Streams, a Amazon GameLift agora oferece um servidor de jogos de alta escala e recursos de streaming de jogabilidade fluida.

[Amazon GameLift Servers](#) oferece aos desenvolvedores de jogos a capacidade de implantar, operar e escalar servidores de jogos dedicados. Você pode implantar servidores de jogos de alto desempenho na nuvem em minutos e escalar para cima e para baixo para atender à demanda dos jogadores. Desenvolvido em um ambiente de computação AWS comprovado, Amazon GameLift Servers suporta 100 milhões de jogadores simultâneos em um único jogo, 100 mil adições de jogadores por segundo e 9 mil novas instâncias de computação por minuto. E com segurança de nível empresarial, matchmaking para as maiores multidões e pay-as-you-go flexibilidade, ele ajuda você a começar, esteja trabalhando em uma nova ideia de jogo ou executando um jogo com milhões de jogadores.

[O Amazon GameLift Streams](#) ajuda os desenvolvedores de jogos a oferecer experiências de streaming de jogos com resolução de até 1080p e 60 frames-per-second (fps) sem atrasos perceptíveis entre dispositivos, incluindo iOS, Android, FireOS e para jogadores. PCs Usando uma única AWS oferta, os editores podem implantar o conteúdo do jogo em minutos, sem modificações, em instâncias de GPU baseadas em nuvem totalmente gerenciadas e entregá-las por meio do AWS Network Backbone diretamente para qualquer dispositivo de jogador final com um navegador da web. Os jogadores podem começar a jogar em apenas alguns segundos, sem esperar por um download ou instalação, e isso oferece uma experiência de jogo quase indistinguível de jogar localmente em um PC ou console de jogos.

14 de janeiro de 2025: Amazon GameLift Servers lança suporte expandido para priorização de posicionamento em sessões de jogos

Versões atualizadas do SDK:

- AWS SDK 1.11.485

Em resposta aos comentários dos clientes, estamos lançando uma nova funcionalidade que permite priorizar locais para solicitações individuais de posicionamento em sessões de jogo. Para suas filas configuradas para priorizar o posicionamento por local, agora você pode fornecer uma lista personalizada de locais prioritários com cada solicitação de posicionamento.

Esse novo recurso permite que os clientes alterem dinamicamente as prioridades de localização para cada solicitação de posicionamento, conforme necessário. A flexibilidade adicional significa que você pode responder melhor às mudanças nas condições, como localização dos jogadores, carga da frota ou integridade do servidor. Ele também pode oferecer suporte a clientes que desejam personalizar ainda mais a forma como os locais de colocação são selecionados.

Saiba mais:

- [Priorize o posicionamento da sessão de jogo, Amazon GameLift Servers Guia](#) do desenvolvedor
- [StartGameSessionPlacement](#), Referência Amazon GameLift Servers da API

2 de janeiro de 2025: Amazon GameLift Servers lança novo suporte para encerrar sessões de jogo

Versões atualizadas do SDK:

- AWS SDK 1.11.477

Em resposta aos comentários dos clientes, estamos lançando uma nova funcionalidade que permite encerrar sessões individuais de jogo com mais facilidade. Com esta versão, agora você pode encerrar uma sessão de jogo diretamente no Amazon GameLift Servers console ou usando o AWS SDK AWS CLI ou para. Amazon GameLift Servers

Esse novo recurso aborda a necessidade de resolver sessões de jogo que permanecem ativas, mas em mau estado, o que impede que os recursos de computação hospedem novas sessões de jogo. Anteriormente, os clientes precisavam acessar remotamente a computação para encerrar manualmente uma sessão de jogo.

Você tem dois métodos de rescisão para escolher. O primeiro método tenta encerrar normalmente uma sessão de jogo usando sua sequência de desligamento personalizada, que pode incluir ações para notificar jogadores e resolver dados do jogo. O segundo método força o processo do servidor a parar, o que encerra a sessão do jogo imediatamente. Esse segundo método garante que a sessão do jogo termine mesmo quando o processo do servidor não estiver respondendo.

Saiba mais:

- [Encerrar uma sessão de jogo usando o Amazon GameLift Servers console](#), Guia Amazon GameLift Servers do desenvolvedor
- [TerminateGameSession](#), Referência Amazon GameLift Servers da API

19 de dezembro de 2024: Amazon GameLift Servers lança suporte ao plug-in do mecanismo de jogo para contêineres gerenciados

Versões atualizadas do plug-in:

Amazon GameLift Serversplugin para Unreal Engine, versão 2.0.0

- Atualizado para oferecer suporte ao SDK 5.2.0 do servidor C++ com suporte a contêineres gerenciados.
- Foi adicionado suporte para Unreal Engine 5.4 e 5.5.

Amazon GameLift Serversplugin para Unity, versão 3.0.0

- Atualizado para oferecer suporte ao SDK 5.2.0 do servidor C++ com suporte a contêineres gerenciados.
- Support para Unity 2021.3 LTS e 2022.3 LTS para Windows e Mac OS.

O Amazon GameLift Servers plug-in para os mecanismos de jogo Unreal e Unity fornece ferramentas e fluxos de trabalho que simplificam suas etapas para colocar um jogo em funcionamento. Amazon GameLift Servers é um serviço de hospedagem em nuvem totalmente gerenciado que os desenvolvedores de jogos podem usar para gerenciar e escalar servidores de jogos dedicados para jogos multijogador baseados em sessões.

As versões mais recentes do plug-in oferecem os seguintes aprimoramentos:

- Fluxo de trabalho guiado para hospedagem com contêineres gerenciados. Esse fluxo de trabalho orienta você nas etapas para configurar uma imagem de contêiner com seu software de servidor de jogos e implantar uma solução de hospedagem baseada em nuvem para seu servidor de jogos. O fluxo de trabalho oferece dois cenários de implantação diferentes: uma implantação simples e uma implantação mais completa com uma fila de posicionamento da sessão de jogo e

um FlexMatch matchmaker. Cada cenário gera frotas de Amazon GameLift Servers contêineres e AWS recursos de apoio.

- Processo aprimorado para configurar perfis de AWS usuário e gerenciar credenciais de AWS acesso para uso de plug-ins. Você pode manter vários perfis para trabalhar com diferentes AWS contas, usuários da conta e regiões.
- Funcionalidade adicional para atualizar as frotas de contêineres existentes. Você pode implantar novas imagens de contêiner (como para atualizações de versões do servidor de jogos) e alterar as configurações da frota sem precisar começar do início.
- Fluxos de trabalho aprimorados para hospedagem com frotas Amazon GameLift Servers em qualquer lugar e gerenciadas. EC2 As melhorias baseadas no feedback dos clientes incluem uma melhor orientação com dicas e links para recursos úteis.

Os cenários de implantação de contêineres gerenciados e EC2 soluções gerenciadas usam AWS CloudFormation modelos para criar e implantar os AWS recursos para cada cenário. Esses modelos estão incluídos no download do Amazon GameLift Servers plug-in e são editáveis. Você pode usá-las como estão ou modificá-las para o seu jogo.

Saiba mais:

- [Plugin para Unreal: implante seu jogo em uma frota de contêineres gerenciada](#), Guia Amazon GameLift Servers do desenvolvedor
- [Plugin para Unity: implante seu jogo em uma frota gerenciada de contêineres](#), Guia Amazon GameLift Servers do desenvolvedor
- [Baixe o plugin para Unreal em GitHub](#)
- [Baixe o plugin para Unity em GitHub](#)
- [Hospedagem de jogos com Amazon GameLift Servers](#)
- [Fórum da Amazon GameLift Servers](#)

12 de novembro de 2024: Amazon GameLift Servers lança hospedagem de jogos multijogador com contêineres gerenciados

Versões atualizadas do SDK:

- AWS SDK 1.11.445
- SDK do servidor, versão 5.2.0 (todos os idiomas)

Amazon GameLift Servers lança para disponibilidade geral uma nova solução de hospedagem para cargas de trabalho de servidores de jogos em contêineres. Com esta versão, os desenvolvedores de jogos agora podem aproveitar os benefícios da containerização, incluindo ambientes consistentes e seguros, um processo de implantação simplificado e utilização otimizada de recursos.

As frotas de contêineres gerenciados usam EC2 instâncias da Amazon que são gerenciadas Amazon GameLift Servers em seu nome e com base em suas configurações. Você cria uma arquitetura de contêiner personalizada para o seu jogo e fornece imagens de contêiner armazenando-as em um repositório do Amazon Elastic Container Registry (Amazon ECR). As frotas de contêineres estão disponíveis somente para servidores de jogos baseados em Linux. Os servidores de jogos devem estar integrados ao Server SDK 5.2.0 ou superior.

Com frotas de contêineres gerenciadas, você obtém os mesmos benefícios das EC2 frotas gerenciadas. Isso inclui suporte para tipos de instância sob demanda e spot, escalabilidade inteligente de capacidade, posicionamento de sessões de jogo com filas e matchmaking. Você também obtém as mesmas métricas de outros tipos de frota, além de algumas novas para contêineres. Outros recursos para frotas de contêineres incluem:

- Alinhamento com a experiência sem servidor para cargas de trabalho em contêineres. Execute um processo de servidor de jogo por contêiner e coloque vários contêineres em cada instância da frota para otimizar o uso dos recursos. Se você preferir ter contêineres com vários processos de servidor de jogos, você pode usar o Amazon GameLift Servers Agente para gerenciamento automatizado do host.
- Criação simplificada de frotas. As frotas de contêineres são projetadas para exigir configurações mínimas de implantação, com valores razoáveis. suggested/default Você pode implantar rapidamente uma frota ativa e, em seguida, personalizar as configurações individuais conforme necessário.
- Ferramentas de controle de versão para arquitetura de contêineres. Agora você pode atualizar uma definição de grupo de contêineres (que é semelhante a uma “tarefa” de contêiner), manter várias versões e especificar qual versão implantar em uma frota.
- Ferramentas de atualização da frota. Com as frotas de contêineres, você não precisa mais criar uma nova frota quando quiser lançar uma atualização da versão do servidor do jogo. Em vez disso, agora você pode atualizar sua imagem de contêiner e implantar as atualizações nas frotas existentes.

Você pode criar frotas de Amazon GameLift Servers contêineres em qualquer Região da AWS lugar que Amazon GameLift Servers ofereça suporte a frotas de vários locais e pode implantar instâncias de frotas de contêineres em qualquer local remoto compatível. Consulte mais detalhes em [Amazon GameLift Servers locais de serviço](#). Atualmente, os contêineres gerenciados não estão disponíveis nas regiões AWS da China.

Saiba mais:

- Publicação no blog: [Aproveite contêineres totalmente gerenciados para hospedar jogos multijogador em escala global em Amazon GameLift Servers](#)
- [Contentores gerenciados](#) visão geral, Guia Amazon GameLift Servers do desenvolvedor
- [Como os contêineres funcionam em Amazon GameLift Servers](#), Guia Amazon GameLift Servers do desenvolvedor
- [Roteiro de desenvolvimento para hospedagem com contêineres Amazon GameLift Servers gerenciados](#), Guia Amazon GameLift Servers do desenvolvedor
- [CreateContainerFleet](#), Referência Amazon GameLift Servers da API

19 de setembro de 2024: Amazon GameLift Servers lança atualização para o SDK do servidor C++ e plug-ins para Unreal Engine

Versões atualizadas do SDK:

SDK do servidor C++, versão 5.1.3

- Novos recursos de registro em log. Agora você pode acessar os logs de solicitação do SDK.
- Confiabilidade aprimorada na transmissão de mensagens do SDK. O SDK agora usa mecanismos de reconexão mais robustos para se recuperar em caso de interrupções na rede ou perdas aleatórias de mensagens.

Versões atualizadas do plug-in:

Amazon GameLift Servers plugin para Unreal Engine, versão 1.1.2

- Atualizado para oferecer suporte à versão mais recente do SDK 5.1.3 do servidor C++.
- No Amazon GameLift Servers plug-in do Unreal Engine, ao procurar um executável de compilação de servidor para uma frota, agora você tem a opção de navegar por todos os arquivos.

Plug-in do SDK do servidor C++ para Unreal, versão 5.1.2

- Atualizado para oferecer suporte à versão mais recente do SDK 5.1.3 do servidor C++.

Saiba mais:

- [Integrando jogos com o Amazon GameLift Servers plug-in do Unreal Engine, Guia do desenvolvedor Amazon GameLift Servers](#)
- [Amazon GameLift Serversdownloads de plug-ins e SDK](#)

5 de setembro de 2024: Amazon GameLift Servers melhora a observabilidade do processo de criação da frota

Com base no feedback dos clientes, esclarecemos o Amazon GameLift Servers fluxo de trabalho para criar uma EC2 frota gerenciada e prepará-la para sediar sessões de jogos. Entre os aprimoramentos estão:

- Fornecemos descrições mais específicas e precisas de cada fase do processo de criação da frota. Essa visibilidade aprimorada facilita a identificação e a resolução de problemas com mais rapidez.
- As fases de criação e ativação separam melhor as tarefas de implantação da instância (criação) das tarefas para iniciar os processos do servidor do jogo e conectar-se ao Amazon GameLift Servers serviço (ativação). Essa alteração facilita o reconhecimento da provável causa dos problemas. Além disso, agora você pode se conectar remotamente às frotas quando elas estão na fase de ativação.
- Dois novos eventos de criação de frota comunicam o êxito ou a falha dos scripts de instalação do servidor de jogos. Se sua compilação de servidor de jogos incluir um script de instalação, Amazon GameLift Servers tentará executar o script e emitirá um dos seguintes novos eventos:
 - FLEET_CREATION_COMPLETED_INSTALLER
 - FLEET_CREATION_FAILED_INSTALLER

Saiba mais:

- [Como funciona a criação de Amazon GameLift Servers frotas](#), Guia Amazon GameLift Servers do desenvolvedor
- [Depurar problemas de frota do Amazon GameLift Servers](#), Guia Amazon GameLift Servers do desenvolvedor

- [Tipo de dados do evento](#), referência Amazon GameLift Servers da API

15 de agosto de 2024: Amazon GameLift Servers melhora a experiência do console

Com base nos comentários dos clientes, fizemos as seguintes atualizações na experiência do [Amazon GameLift Serversconsole](#):

- Suas preferências de exibição de páginas agora são salvas automaticamente na sua AWS conta de usuário e aplicadas sempre que você retorna à página. As preferências de exibição permitem que você escolha quais informações incluir em uma exibição de tabela, como na página de listagem de frotas. Personalize suas preferências de exibição usando o ícone  no canto superior direito de uma tabela.
- O fluxo de trabalho Criar frota para EC2 frotas gerenciadas foi simplificado para combinar a seleção de locais da frota e tipos de instância. Facilitamos a localização do tipo de instância certo para a frota, mesmo quando você altera as seleções de locais.

Saiba mais:

- [Crie uma EC2 frota Amazon GameLift Servers gerenciada](#), Guia Amazon GameLift Servers do desenvolvedor

25 de julho de 2024: Amazon GameLift Servers adiciona suporte para a Zona Local da AWS Nigéria

Com a hospedagem Amazon GameLift Servers gerenciada, agora você pode implantar recursos de servidores de jogos na Nigéria, na África Ocidental e estender o alcance de seus jogos a jogadores de toda a África. Use as Zonas AWS Locais para colocar os servidores de jogos geograficamente mais próximos de seus jogadores para reduzir a latência e melhorar significativamente a jogabilidade.

Para começar imediatamente a hospedar sessões de jogos na Nigéria, adicione a nova zona local da Nigéria como um local remoto a uma frota nova ou existente em vários locais. Se o seu jogo usa Amazon GameLift ServersFlexMatch, atualize as frotas em sua fila de matchmaking para incluir a nova Zona Local. Com frotas de vários locais, você pode gerenciar diretamente a capacidade de hospedagem em cada local.

A matriz Região da AWS da Zona Local de Lagos, Nigéria, é a Região da África (Cidade do Cabo) af-south-1 (), Amazon GameLift Servers que também oferece suporte como local remoto. O nome da zona local da Nigéria é af-south-1-los-1.

Saiba mais:

- [Amazon GameLift Servers locais de serviço](#), Guia Amazon GameLift Servers do desenvolvedor
- [Atualizar localizações da frota](#), Guia Amazon GameLift Servers do desenvolvedor

02 de julho de 2024: Amazon GameLift Servers lança nova ferramenta de console para visualizar dados da sessão do jogador

O Amazon GameLift Servers console agora oferece uma ferramenta de pesquisa de sessão de jogador que permite recuperar informações da sessão do jogador por ID da sessão do jogo, ID da sessão do jogador ou ID do jogador. Jogos que usam FlexMatch matchmaking geram automaticamente sessões de jogadores para cada jogador correspondente. Para todos os outros jogos, as sessões de jogadores são um recurso opcional.

Você pode encontrar a ferramenta de pesquisa de sessão do jogador na navegação principal do Amazon GameLift Servers console. Visualize as sessões individuais de jogadores ou compare dados em sessões de vários jogadores. Você também pode abrir os dados da sessão de jogador ao visualizar a página de detalhes da sessão do jogo.

Saiba mais:

- [Sessões de jogos e jogadores no Amazon GameLift Servers console](#), Guia Amazon GameLift Servers do desenvolvedor

24 de abril de 2024: Amazon GameLift Servers lança prévia de frotas de contêineres

Amazon GameLift Servers agora oferece uma prévia das frotas de contêineres, que oferecem maior portabilidade, escalabilidade, tolerância a falhas e agilidade.

Em frotas de contêineres, as EC2 instâncias da Amazon hospedam um ou mais de seus contêineres. Esses contêineres incluem o servidor de jogos, juntamente com tudo o que for necessário, incluindo dependências e configurações. Exemplos de dependências incluem pacotes SDKs de software. Depois de carregar seu contêiner para o Amazon Elastic Container Registry privado, Amazon GameLift Servers preenche sua frota com o contêiner.

Para funcionar em uma frota de contêineres, o servidor de jogos deve ser executado em Linux e estar integrado ao SDK 5.x do servidor. Em uma frota de contêineres, você tem um controle ajustado dos recursos de hospedagem, permitindo otimizar o consumo de recursos, como unidades de CPU e memória. Você também pode hospedar vários servidores de jogos em um contêiner para reduzir o uso de recursos.

Em uma frota de contêineres, você obtém muitos dos mesmos benefícios que outros tipos de frotas possuem, como tipos de instância sob demanda, ajuste de escala (automático e manual), filas e criação de parcerias. Você também obtém as mesmas métricas de outros tipos de frota, além de algumas novas para contêineres. As frotas de contêineres oferecem alcance global aos jogadores nos seguintes locais e regiões:

- ap-northeast-1
- ap-northeast-2
- ap-southeast-2
- eu-central-1
- eu-west-1
- us-east-1
- us-west-2

Para alcançar ainda mais regiões e zonas locais, crie frotas de contêineres com vários locais.

Saiba mais:

- [Gerenciando hospedagem com Amazon GameLift Servers contêineres](#), Guia Amazon GameLift Servers do desenvolvedor
- [CreateContainerGroupDefinition](#), Referência Amazon GameLift Servers da API

13 de fevereiro de 2024: Amazon GameLift Servers lança melhorias e simplifica a SDKs instalação do Amazon GameLift Servers plug-in para o Unreal Engine

Versões atualizadas do SDK:

- SDK do servidor Go, versão 5.1.0
- SDK do servidor C#, versão 5.1.2
- SDK do servidor C++, versão 5.1.2

Fizemos as seguintes melhorias:

- Melhoramos a confiabilidade do SDK adicionando reconexão automática em caso de interrupção da rede.
- [Go] Agora você pode chamar `InitSDK()` com ou sem parâmetros do servidor. Os servidores de jogos Amazon GameLift Servers executados em EC2 frotas gerenciadas leem os parâmetros do servidor diretamente das variáveis de ambiente. Os servidores de jogos nas frotas do Amazon GameLift Servers Anywhere devem ligar `InitSDK()` com os parâmetros do servidor.

Versões atualizadas do plug-in:

- Amazon GameLift Serversplugin para Unreal Engine, versão 1.1.0
- Amazon GameLift Serversplugin para Unity, versão 2.1.0
- Plug-in do SDK do servidor C++ para Unreal, versão 5.1.1
- Plug-in do SDK do servidor C# para Unity, versão 5.1.2

Fizemos as seguintes melhorias:

- [Amazon GameLift Serversplugin para Unreal Engine] Atualizou as instruções de instalação e simplificou o empacotamento. Esse plug-in agora inclui a versão mais recente do SDK do servidor C++ para Unreal.
- Atualizou os plug-ins para oferecer suporte à versão mais recente do SDK do servidor para Amazon GameLift Servers

Saiba mais:

- [Integrando jogos com o Amazon GameLift Servers plug-in do Unreal Engine, Guia do desenvolvedor Amazon GameLift Servers](#)
- [Amazon GameLift Serversdownloads de plug-ins e SDK](#)

14 de dezembro de 2023: Amazon GameLift Servers adiciona a capacidade de atualizar as propriedades do jogo das sessões ativas do jogo

Você já conseguiu definir as propriedades do jogo ao criar sessões de jogo e pesquisar as propriedades especificadas nas sessões de jogo. Agora você também pode adicionar e atualizar essas propriedades em uma sessão de jogo ativa.

Por exemplo, os jogadores votam em um mapa no qual querem jogar. O cliente de jogo chama `UpdateGameSession` para modificar um valor `GameProperty` para `{"Key": "map", "Value": "jungle"}`. O jogo implementa o novo mapa para os jogadores na sessão de jogo.

Os administradores do jogo também podem recuperar dados úteis das propriedades do jogo usando a operação `SearchGameSessions`. Por exemplo, os administradores podem listar sessões de jogo que têm um valor de `Status` como `ACTIVE` e esta propriedade do jogo: `{"Key": "map", "Value": "desert"}`.

Saiba mais:

- [the section called “Adicionar Amazon GameLift Servers a um cliente de jogo”](#), Guia Amazon GameLift Servers do desenvolvedor
- [GameProperty](#), Referência Amazon GameLift Servers da API
- [UpdateGameSession](#), Referência Amazon GameLift Servers da API
- [SearchGameSessions](#), Referência Amazon GameLift Servers da API

21 de novembro de 2023: Amazon GameLift Servers lança suporte para ferramentas de infraestrutura como código, como Terraform e Pulumi, desenvolvidas por AWS API Cloud Control

Agora você pode gerenciar toda a pilha de Amazon GameLift Servers recursos usando as ferramentas de Infraestrutura como Código (IaC). Essas ferramentas incluem AWS CloudFormation, e também, ferramentas de terceiros, como Terraform e Pulumi. Com esse suporte adicional, agora você pode se concentrar em criar seu jogo e aproveitar DevOps estratégias para cuidar do gerenciamento de recursos, CI/CD e implantação para seus clientes.

Agora você também pode provisionar e configurar todos os tipos de Amazon GameLift Servers recursos usando a AWS Cloud Control API. Você pode continuar trabalhando com recursos usando os Amazon GameLift Servers APIs ou os AWS CloudFormation modelos do Amazon GameLift Servers.

Para obter detalhes sobre os Amazon GameLift Servers recursos disponíveis por meio do IaC, consulte a [Amazon GameLift Serversreferência](#) do tipo de Amazon GameLift Servers recurso.

Além disso, agora você pode escalar automaticamente suas frotas usando AWS CloudFormation modelos ou a API AWS Cloud Control usando a nova propriedade [Fleet:ScalingPolicies](#).

A API Cloud Control oferece aos desenvolvedores um conjunto padrão APIs para criar, ler, atualizar, excluir e listar recursos (CRUDL) em centenas de AWS serviços e várias ferramentas de terceiros, como Terraform e Pulumi.

Saiba mais:

- [AWS CloudFormation](#)
- [AWS API de controle de nuvem](#)
- [AWS Fornecedor CC Terraform](#)
- [Pulumi](#)

16 de novembro de 2023: Amazon GameLift Servers atualiza o plug-in autônomo para Unity

Versões atualizadas do SDK: Amazon GameLift Servers plugin para Unity, versão 2.0.0

O Amazon GameLift Servers plug-in para Unity fornece ferramentas e fluxos de trabalho que simplificam as etapas para colocar seu jogo Unity em funcionamento para hospedagem na nuvem com. Amazon GameLift Servers Amazon GameLift Serversé um serviço totalmente gerenciado que permite aos desenvolvedores de jogos gerenciar e escalar servidores de jogos dedicados para jogos multijogador baseados em sessões.

Com esta versão, o plug-in para Unity é atualizado para usar os Amazon GameLift Servers recursos mais recentes, incluindo o SDK do servidor versão 5.x e suporte para testes locais com Amazon GameLift Servers o Anywhere. O plug-in é compatível com as versões do Unity 2021.3 LTS e 2022.3 LTS.

Os principais recursos do plug-in incluem:

- Fluxos de trabalho de UI guiados no Editor Unity para os seguintes cenários:
 - Teste sua integração de jogos Amazon GameLift Servers usando sua estação de trabalho local como host. Esse fluxo de trabalho ajuda você a configurar uma frota Amazon GameLift Servers

Anywhere para sua máquina local, iniciar instâncias do seu servidor e cliente de jogos, solicitar uma sessão de Amazon GameLift Servers jogo e entrar no jogo.

- Implemente uma solução de hospedagem em nuvem para seu servidor de jogos integrado com AWS recursos Amazon GameLift Servers gerenciados EC2 e de suporte. Esse fluxo de trabalho ajuda você a configurar o jogo para hospedagem na nuvem e fornece três cenários de implantação:
 - Implante o servidor de jogos em uma única frota.
 - Implante o servidor do jogo em um conjunto de frotas spot de baixo custo em várias AWS regiões.
 - Implante o servidor do jogo com um FlexMatch matchmaker.
- Capacidade de configurar perfis de usuário vinculados a um usuário AWS da conta e definir uma AWS região padrão. Você pode manter vários perfis para trabalhar em diferentes AWS contas, usuários da conta e regiões.
- Conveniências especiais que ajudam a agilizar os processos de Amazon GameLift Servers integração e implantação, incluindo:
 - Cada solução de hospedagem inclui AWS recursos de suporte, incluindo um grupo de usuários do Amazon Cognito que fornece validação exclusiva para jogadores IDs e jogadores. As soluções também incluem um bucket do Amazon S3 para armazenamento, notificação de eventos do Amazon SNS AWS Lambda , funções e outros recursos.
 - Para o fluxo de trabalho Anywhere, o plug-in automatiza as configurações de parâmetros do servidor necessárias.
 - Para o EC2 fluxo de trabalho da Amazon, cada solução de implantação fornece um serviço de back-end de cliente integrado usando funções Lambda. O serviço de back-end fica entre o cliente do jogo e o Amazon GameLift Servers serviço e gerencia todas as chamadas diretas para o Amazon GameLift Servers serviço.
- Conteúdo para testes de integração, incluindo ativos e código para um simples exemplo de jogo multijogador para ilustrar a integração entre servidor e cliente de jogo.
- Documentação do plug-in com orientação detalhada de integração e código de amostra.

Todos os cenários de implantação, inclusive para EC2 frotas da Anywhere e da Amazon, usam AWS CloudFormation modelos para descrever e implantar os AWS recursos para a solução do seu jogo. Esses modelos estão incluídos no download do Amazon GameLift Servers plugin. Você pode usá-los como estão ou personalizá-los para o jogo.

Saiba mais:

- [Amazon GameLift Serversplugin para Unity \(servidor SDK 5.x\)](#), Guia Amazon GameLift Servers do desenvolvedor
- [Baixe o plugin em GitHub](#)
- [Sobre Amazon GameLift Servers hospedagem](#)
- [Fórum da Amazon GameLift Servers](#)

2 de novembro de 2023: Amazon GameLift Servers adiciona suporte para credenciais compartilhadas

Versões atualizadas do SDK: AWS SDK 1.11.193

O novo recurso de credenciais Amazon GameLift Servers compartilhadas permite que aplicativos implantados em EC2 frotas gerenciadas interajam com outros recursos. AWS Essa atualização afeta aplicações agrupadas e implantadas juntamente com os binários do servidor de jogos integrados ao SDK versão 5.x ou posterior do servidor. (Os executáveis do servidor de jogos já podem solicitar credenciais usando a ação `GetFleetRoleCredentials()` 5.x do SDK de servidor.)

Por exemplo, se você quiser implantar sua compilação de servidor de jogos com um CloudWatch agente da Amazon para coletar métricas de EC2 instância e outros dados, o agente precisa de permissão para interagir com seus CloudWatch recursos. Para fazer isso, você deve primeiro configurar uma função (AWS Identity and Access Management IAM) com permissões para usar os CloudWatch recursos e, em seguida, configurar uma frota com a função do IAM e as credenciais compartilhadas ativadas. Quando Amazon GameLift Servers implanta sua compilação de servidor de jogos em cada EC2 instância, ele gera um arquivo de credenciais compartilhado e o armazena na instância. Todos os aplicativos na instância podem usar as credenciais compartilhadas. Amazon GameLift Serversatualiza automaticamente as credenciais temporárias durante toda a vida útil da instância.

Você pode ativar as credenciais compartilhadas ao criar uma EC2 frota gerenciada usando os seguintes métodos:

- No fluxo de trabalho de criação de frotas do Amazon GameLift Servers console.
- Ao chamar a operação da API de serviço `CreateFleet` usando o novo parâmetro `InstanceRoleCredentialsProvider`.

- Ao chamar a operação AWS CLI `aws gamelift create-fleet` com o parâmetro `instance-role-credentials-provider`

Saiba mais:

- [Comunique-se com outros AWS recursos de suas frotas](#), Guia Amazon GameLift Servers do desenvolvedor
- [CreateFleet, InstanceRoleCredentialsProvider](#), Referência Amazon GameLift Servers da API
- [Configurar uma função de serviço do IAM](#), Guia Amazon GameLift Servers do desenvolvedor

28 de setembro de 2023: Amazon GameLift Servers lança novo plug-in independente para o Unreal Engine

Versões atualizadas do SDK: Amazon GameLift Servers plug-in para Unreal Engine versão 1.0.0

O Amazon GameLift Servers plug-in do Unreal Engine fornece ferramentas e fluxos de trabalho que simplificam suas etapas para colocar um jogo em funcionamento Amazon GameLift Servers para hospedagem na nuvem. Amazon GameLift Servers é um serviço totalmente gerenciado que permite aos desenvolvedores de jogos gerenciar e escalar servidores de jogos dedicados para jogos multijogador baseados em sessões. O plug-in suporta as versões 5.0, 5.1 e 5.2 da UE. Os principais recursos incluem:

- Fluxos de trabalho de UI guiados no editor Unreal percorrem os seguintes caminhos:
 - Teste sua integração de jogos Amazon GameLift Servers usando sua estação de trabalho local como host. Esse fluxo de trabalho ajuda você a configurar uma frota Amazon GameLift Servers Anywhere para sua máquina local, iniciar instâncias do seu servidor e cliente de jogos, solicitar uma sessão de Amazon GameLift Servers jogo e obter informações de conexão para a nova sessão de jogo.
 - Implante uma solução de hospedagem EC2 na nuvem da Amazon para seu servidor de jogos integrado. Esse fluxo de trabalho ajuda você a configurar seu jogo para hospedagem na nuvem e fornece três cenários de implantação diferentes: implante em uma única frota, implante em um conjunto de frotas spot em várias regiões ou implante em um conjunto de frotas com um FlexMatch matchmaker. A solução para cada cenário de implantação inclui Amazon GameLift Servers recursos e recursos de suporte AWS .

- Capacidade de configurar perfis de usuário vinculados a um usuário AWS da conta e definir uma AWS região padrão. Você pode manter vários perfis para trabalhar em diferentes AWS contas, usuários da conta e regiões.
- Conveniências especiais que ajudam a agilizar os processos de Amazon GameLift Servers integração e implantação, incluindo:
 - Cada solução de hospedagem inclui AWS recursos de suporte, incluindo um grupo de usuários básico do Amazon Cognito que fornece um player exclusivo IDs, um bucket do Amazon S3 para armazenamento, notificação de eventos e funções do Amazon SNS. AWS Lambda
 - Para o fluxo de trabalho Anywhere, o plug-in automatiza as configurações de parâmetros do servidor necessárias usando argumentos de linha de comando.
 - Para o EC2 fluxo de trabalho da Amazon, cada solução de implantação fornece um serviço de back-end de cliente integrado usando funções Lambda. O serviço de back-end recebe solicitações de clientes de jogos e as repassa para o Amazon GameLift Servers serviço.
- Conteúdo para testes de integração, incluindo um mapa inicial do jogo e dois mapas de teste com esquemas básicos e elementos de interface do usuário.
- Documentação do plug-in com orientação detalhada de integração e código de amostra.

Todos os cenários de implantação, inclusive para EC2 frotas da Anywhere e da Amazon, usam AWS CloudFormation modelos para descrever as soluções. O plug-in usa esses modelos ao implantar Amazon GameLift Servers recursos para seu jogo. Esses modelos estão incluídos no download do Amazon GameLift Servers plug-in e são editáveis. Você pode usá-las como estão ou modificá-las para o seu jogo.

Saiba mais:

- [Amazon GameLift Serversplugin para Unreal Engine](#), Guia Amazon GameLift Servers do desenvolvedor
- [Baixe o plugin em GitHub](#)
- [Sobre Amazon GameLift Servers hospedagem](#)
- [Fórum da Amazon GameLift Servers](#)

17 de agosto de 2023: Amazon GameLift Servers oferece hospedagem de servidores de jogos com processadores AWS Graviton

Versões atualizadas do SDK: AWS SDK 1.11.144

Com Amazon GameLift Servers isso, agora você pode hospedar seus jogos na nuvem usando EC2 instâncias com processadores AWS Graviton. Projetadas AWS com processadores baseados em ARM64, as instâncias Graviton oferecem a melhor relação preço/desempenho para o uso de cargas de trabalho em nuvem EC2, com até 40% de melhoria em relação às instâncias comparáveis baseadas em x86. Os processadores Graviton3 mais recentes oferecem desempenho computacional até 25% melhor em relação às versões anteriores.

Com Amazon GameLift Servers, agora você pode selecionar entre essas novas instâncias da família AWS Graviton:

- Instâncias baseadas em Graviton2: c6g, c6gn, r6g, m6g, g5g
- Instâncias baseadas em Graviton3: c7g, r7g, m7g

Saiba mais:

- [AWS Processador Graviton](#): saiba mais sobre os benefícios e os usos práticos das instâncias baseadas em EC2 Graviton.
- [Introdução ao Graviton](#): tenha uma visão geral das instâncias baseadas em Graviton e insights sobre como as aplicações são executadas nelas, dependendo do sistema operacional, dos idiomas e dos tempos de execução.

Note

As instâncias do Graviton Arm exigem um Amazon GameLift Servers servidor baseado no sistema operacional Linux. O Server SDK 5.1.1 ou mais recente é necessário para C++ e C#. O Server SDK 5.0 ou mais recente é necessário para o Go. Essas instâncias não oferecem out-of-the-box suporte para a instalação do Mono no Amazon Linux 2023 (AL2023) ou no Amazon Linux (AL22).

27 de julho de 2023: Amazon GameLift Servers lança o SDK 5.1.0 do servidor com suporte adicional para desenvolvimento em Unity

Versões atualizadas do SDK: SDK de servidor para C++, C#/Unity, Unreal 5.1.0

A versão mais recente do SDK do Amazon GameLift Servers servidor oferece atualizações para C++, C# e o plug-in Unreal, além de um novo plug-in para uso com o motor de jogo Unity. Os

desenvolvedores de jogos integram o SDK Amazon GameLift Servers do servidor aos servidores de jogos nos quais eles implantam para hospedagem. Amazon GameLift Servers

A versão mais recente do SDK de servidor contém as seguintes atualizações, que incluem várias solicitações de clientes:

- Baixe pacotes SDK específicos do idioma — O [site de Amazon GameLift Servers download](#) atualizado contém pacotes SDK para cada idioma. Você pode baixar as versões atuais ou anteriores.
- Novo plug-in de SDK de servidor C# para Unity — O novo pacote de SDK de servidor para Unity contém bibliotecas C# criadas que você pode instalar usando o gerenciador de pacotes no Unity Editor (consulte o novo guia de integração do [Unity](#)). Essas bibliotecas incluem as dependências necessárias por meio de UnityNuGet. Você pode usar esse plug-in com o Unity 2020.3 LTS, 2021.3 LTS e 2022.3 LTS para Windows e Mac OS. Ele suporta os perfis .NET Framework e .NET Standard do Unity, com o .NET Standard 2.1 e .NET 4.x.
- Solução .NET consolidada para C#: O SDK de servidor para C# agora oferece suporte a o .NET Framework 4.6.2 (atualizado da versão 4.6.1) e o .NET 6.0 em uma única solução. O .NET Standard 2.1 está disponível com as bibliotecas criadas pelo Unity.
- Atualizações do Server SDK 5.1.0
 - [C++, C#, Unreal] Agora você pode chamar `InitSDK()` com ou sem parâmetros do servidor. Os servidores de jogos Amazon GameLift Servers executados em EC2 frotas gerenciadas leem os parâmetros do servidor diretamente das variáveis de ambiente. Os servidores de jogos nas frotas do Amazon GameLift Servers Anywhere devem ligar `InitSDK()` com os parâmetros do servidor.
 - [C++, C#, Unreal] As chamadas do SDK de servidor melhoraram as mensagens de erro.
 - [SDK para C++] Para melhorar os tempos de compilação do SDK de servidor, o sinalizador de compilação `-DRUN_CLANG_FORMAT` fica desativado por padrão. Você pode habilitá-lo com `-DRUN_CLANG_FORMAT=1`.
 - [SDK para C++] Ao criar as bibliotecas sem as bibliotecas padrão (`-DGAMELIFT_USE_STD=0`), `InitSDK()` não usa mais tipos de `std::` dados.
- Documentação expandida do SDK 5.x para servidor
 - Guias de referência do SDK de servidor atualizados para C++, C#/Unity e Unreal, incluindo cobertura expandida de todos os tipos de dados.
 - [SDK 5.x do servidor C# para Amazon GameLift Servers -- Ações](#)
 - [SDK 5.x do servidor C++ para Amazon GameLift Servers -- Ações](#)

- [SDK 5.x do servidor C++ \(Unreal\) para -- Ações Amazon GameLift Servers](#)
- Novas versões dos guias de integração do SDK 5 do servidor para plug-ins Unity e Unreal
 - [Amazon GameLift ServersIntegre-se a um projeto Unity](#)
 - [Amazon GameLift ServersIntegre-se a um projeto Unreal Engine](#)
- Atualizações adicionais da documentação
 - Documentação revisada para operações Amazon GameLift Servers de API de serviço [GetComputeAccess](#) e [GetInstanceAccess](#) para esclarecer os procedimentos de acesso remoto com base na versão do SDK Amazon GameLift Servers do servidor em uso.
 - Descrições revisadas [GameSessionPlacement](#) para documentar como as informações da sessão de jogo são transitórias quando uma colocação está no status “pendente”.

13 de julho de 2023: Amazon GameLift Servers adiciona métricas de hardware da frota

Agora você pode monitorar as métricas de desempenho de hardware de suas EC2 frotas Amazon GameLift Servers gerenciadas. As métricas incluem métricas de EC2 instância para utilização da CPU, volume de tráfego de rede e read/write atividade de disco. Pois Amazon GameLift Servers, essas métricas descrevem todas as instâncias ativas em um local de frota. Você pode visualizar essas métricas de hardware da frota usando um CloudWatch painel da Amazon no AWS Management Console. Você também pode visualizá-los no Amazon GameLift Servers console nos detalhes da frota.

Saiba mais:

- [Monitore Amazon GameLift Servers com a Amazon CloudWatch](#) (Métricas para frotas), Guia Amazon GameLift Servers do desenvolvedor

29 de junho de 2023: Amazon GameLift Servers lança suporte para Amazon Linux 2023

Versões atualizadas do SDK: AWS SDK 1.11.111

Amazon GameLift Servers clientes agora podem usar o sistema operacional Amazon Linux 2023 para hospedar seus servidores de jogos. AL2023 oferece várias melhorias em relação à AL2 inclusão da segurança. Esse sistema operacional está disponível em todos, Regiões da AWS com exceção das regiões da China.

Os clientes podem usar os sistemas operacionais Linux mais recentes e continuar recebendo atualizações críticas de segurança quando o suporte ao Amazon Linux (AL1) terminar em dezembro de 2023. O suporte para o Amazon Linux 2 continua até 30 de junho de 2025.

Saiba mais:

- [Amazon GameLift Servers Perguntas frequentes sobre o servidor Linux](#)
- [Comparação entre Amazon Linux 2 e Amazon Linux 2023](#)
- Amazon GameLift Servers Links de referência da API:
 - [AWS Ação do SDK CreateBuild](#)
 - [Comando da CLI upload-build](#)
 - [Comando da CLI create-build](#)

25 de maio de 2023: Amazon GameLift Servers FleetIQ adiciona filtro para excluir posicionamentos de sessões de jogo em instâncias de drenagem

Versões atualizadas do SDK: AWS SDK 1.11.87

Se você usa Amazon GameLift Servers FleetIQ para hospedagem de jogos, agora você pode evitar posicionamentos de sessões de jogo em instâncias que estão se esgotando no momento. As instâncias de drenagem estão sinalizadas para encerramento, mas elas ainda podem ser selecionadas para hospedar novas sessões de jogo se nenhum outro recurso de hospedagem estiver disponível. Com esse novo atributo, você pode excluir totalmente o uso de instâncias de drenagem.

Use esse atributo ao ligar `ClaimGameServer` para encontrar servidores de jogos disponíveis. Adicione o novo `FilterOption` parâmetro e defina os status de instância permitidos somente como ATIVO. Em resposta, Amazon GameLift Servers FleetIQ analisa apenas as instâncias ativas ao pesquisar e reivindicar um servidor de jogo disponível.

Saiba mais:

- [ClaimGameServer](#) na Referência de API do Amazon GameLift Servers
- [Como FleetIQ funciona](#) o Guia do Amazon GameLift Servers FleetIQ Desenvolvedor

16 de maio de 2023: Amazon GameLift Servers suporta a etiquetagem de alocação de custos para frotas

Amazon GameLift Servers Agora, os clientes podem usar tags de alocação de AWS Billing custos para organizar seus custos de hospedagem de jogos. Você pode atribuir etiquetas de alocação de custos aos recursos individuais Amazon GameLift Servers EC2 da frota para monitorar como suas frotas estão contribuindo para os custos gerais de hospedagem.

Saiba mais:

- [Gerencie seu Amazon GameLift Servers custos de hospedagem](#)
- [Uso de tags de alocação de custos AWS](#) no AWS Billing Guia do usuário.

20 de abril de 2023: Amazon GameLift Servers lança suporte para o Windows Server 2016

Versões atualizadas do SDK: AWS SDK 1.11.63

Amazon GameLift Servers Os clientes agora podem usar o sistema operacional Windows Server 2016 para hospedar seus servidores de jogos. Esse sistema operacional está disponível em todos Regiões da AWS. Os clientes podem usar o sistema operacional Windows mais recente e continuar recebendo atualizações críticas de segurança à medida que a Microsoft encerra seu suporte para o Windows Server 2012 em outubro de 2023.

A partir de hoje, novos clientes que precisam de um ambiente de runtime do Windows devem especificar o Windows Server 2016 ao criar novas versões de servidores de jogos para hospedagem. Os clientes existentes podem continuar criando novas versões e frotas com o Windows Server 2012, mas devem concluir a migração com o Windows Server 2016 antes da data de fim do suporte da Microsoft, em 10 de outubro de 2023.

Esta atualização inclui as seguintes alterações de serviço:

- Ao criar uma compilação de servidor de jogos usando comandos Amazon GameLift Servers SDK ou CLI, agora você deve definir explicitamente o sistema operacional. Não há valor padrão. Para implantar seu servidor de jogos no Windows Server 2016, você pode usar o valor `WINDOWS_2016`.
- Ao criar uma compilação de servidor de jogos usando o Amazon GameLift Servers console, você deve selecionar um sistema operacional a partir dos valores disponíveis. Se você já é um cliente com frotas ativas do Windows Server 2012, você pode escolher entre uma e `WINDOWS_2012` outra `WINDOWS_2016` opção.

Saiba mais:

- Amazon GameLift Servers [Links de referência da API](#):
 - [Comando da CLI upload-build](#)
 - [Comando da CLI create-build](#)
 - [AWS Ação do SDK CreateBuild](#)
- [Amazon GameLift Servers Perguntas frequentes sobre o Windows 2012](#)

13 de abril de 2023: Amazon GameLift Servers lança o SDK 5.x do servidor para Unreal

Versões atualizadas do SDK: SDK 5.0.0 de servidor para Unreal

A versão mais recente do plug-in Amazon GameLift Servers leve para o Unreal Engine agora é baseada no SDK Amazon GameLift Servers 5.x do servidor. Para começar a integrar seu ambiente Unreal Engine, Amazon GameLift Servers consulte os links a seguir.

Saiba mais:

- [Amazon GameLift Servers Integre-se a um projeto Unreal Engine](#)
- [Adicione Amazon GameLift Servers ao seu servidor de jogos](#)
- [SDK 5.x do servidor C++ para Amazon GameLift Servers -- Ações](#)

14 de março de 2023: Amazon GameLift Servers lança uma nova experiência de console

O novo Amazon GameLift Servers console inclui essas melhorias:

- Navegação aprimorada — O novo painel de navegação facilita a navegação entre Amazon GameLift Servers recursos.
- Amazon GameLift Servers página inicial — A nova página inicial fornece links para documentação útil, exibe uma visão geral de Amazon GameLift Servers alto nível e fornece suporte por meio de links para documentação, perguntas frequentes e AWS re:Post
- CloudWatch Métricas aprimoradas da Amazon — as Amazon GameLift Servers métricas agora estão disponíveis no Amazon GameLift Servers console e em seus CloudWatch painéis. Essa atualização também inclui novas métricas de desempenho, utilização e sessões de jogadores.

Saiba mais:

- [Acompanhe a hospedagem de jogos no Amazon GameLift Servers console](#)
- [Construindo um FlexMatch casamenteiro](#)

14 de fevereiro de 2023: Amazon GameLift Servers agora oferece suporte à criptografia do lado do servidor para tópicos do Amazon SNS

A criptografia do lado do servidor (SSE) para tópicos do SNS criptografa seus dados confidenciais em repouso. SSE usa as teclas AWS Key Management Service (AWS KMS) para proteger o conteúdo de seus tópicos do SNS.

Saiba mais:

- [Configurar notificações de eventos para o posicionamento de sessões de jogo](#)
- [FlexMatch eventos de matchmaking](#)
- [Criptografia em repouso](#)

9 de fevereiro de 2023: o SDK Amazon GameLift Servers do servidor é compatível com o.NET 6 com C #10

Versões atualizadas do SDK: SDK 5.0.0 de servidor para .NET 6 Nenhuma atualização do SDK é necessária.

Se você usa a Unity Real-Time Development Platform, continue usando o SDK 5.0.0 Amazon GameLift Servers do servidor com o.NET 4.6. O Unity não é compatível com o.NET 6.

Saiba mais:

- Baixe a versão mais recente do SDK Amazon GameLift Servers do servidor em [Amazon GameLift ServersIntrodução](#)
- [SDK 5.x do servidor C# para Amazon GameLift Servers -- Ações](#)

31 de janeiro de 2023: o SDK Amazon GameLift Servers do servidor é compatível com a linguagem Go

Versões atualizadas do SDK: SDK 5.0.0 de servidor para Go

Saiba mais:

- Baixe a versão mais recente do SDK Amazon GameLift Servers do servidor em [Amazon GameLift ServersIntrodução](#)
- [SDK do servidor Go para Amazon GameLift Servers -- Ações](#)

1º de dezembro de 2022: Amazon GameLift Servers lança o Amazon GameLift Servers Anywhere and Amazon GameLift Servers Server SDK 5.0

Versões atualizadas do SDK: AWS SDK 1.10.21, Server SDK 5.0.0 para C++ e C#

Amazon GameLift ServersAnywhere usa os recursos do seu servidor de jogos para hospedar servidores de Amazon GameLift Servers jogos. Você pode usar o Amazon GameLift Servers Anywhere para integrar seus próprios recursos computacionais à computação Amazon GameLift Servers EC2 gerenciada para distribuir seus servidores de jogos em vários tipos de computação. Você também pode usar o Amazon GameLift Servers Anywhere para testar iterativamente seus servidores de jogos sem fazer o upload da compilação Amazon GameLift Servers para cada iteração.

Destaques:

- Novos tipos de frota e computação em Amazon GameLift Servers qualquer lugar
- Amazon GameLift ServersRegistro de recursos computacionais em qualquer lugar
- Ciclo de iteração de teste aprimorado

Amazon GameLift ServersO Server SDK 5.0.0 introduz melhorias no SDK do servidor existente e em um novo tipo de recurso, a computação. O Server SDK 5.0.0 oferece suporte a Amazon GameLift Servers Anywhere e ao uso de seus próprios recursos computacionais para hospedagem de servidores de jogos.

Saiba mais:

- [SDK de servidor 5.x para Amazon GameLift Servers](#)
- [Localização da frota](#)
- [Escolher recursos de computação para uma frota gerenciada](#)
- [Crie um Amazon GameLift Servers Frota em qualquer lugar](#)

25 de agosto de 2022: Amazon GameLift Servers lança suporte para Locais Zones

Versões atualizadas do SDK: AWS SDK 1.9.333

Amazon GameLift Servers agora está disponível em oito Zonas Locais nos Estados Unidos da América, para que você possa posicionar suas frotas mais perto dos jogadores. Você pode usar todos os Amazon GameLift Servers recursos gerenciados com Zonas Locais adicionando as Zonas Locais às suas frotas.

As Zonas AWS Locais estendem recursos e serviços até a borda da nuvem, perto de grandes centros populacionais, industriais e de tecnologia da informação (TI). Isso significa que é possível implantar aplicativos que exigem latência de um dígito em milissegundos mais perto dos usuários finais ou dos data centers on-premises.

Saiba mais:

- [Amazon GameLift Servers zonas locais](#)
- [Localização da frota](#)
- [Crie uma EC2 frota Amazon GameLift Servers gerenciada](#)

28 de junho de 2022: Amazon GameLift Servers lança uma nova experiência de console opcional

O novo Amazon GameLift Servers console inclui essas melhorias:

- Navegação aprimorada — O novo painel de navegação facilita a navegação entre Amazon GameLift Servers recursos.
- Amazon GameLift Servers página inicial — A nova página inicial fornece links para documentação útil, exibe uma visão geral de Amazon GameLift Servers alto nível e fornece suporte por meio de links para documentação, perguntas frequentes e AWS re:Post
- CloudWatch Métricas aprimoradas da Amazon — as Amazon GameLift Servers métricas agora estão disponíveis no Amazon GameLift Servers console e em seus CloudWatch painéis. Essa atualização também inclui novas métricas de desempenho, utilização e sessões de jogadores.

Saiba mais:

- [Acompanhe a hospedagem de jogos no Amazon GameLift Servers console](#)

- [Construindo um FlexMatch casamenteiro](#)

15 de fevereiro de 2022: FlexMatch adiciona regra composta e melhorias adicionais

FlexMatches usuários agora têm acesso aos seguintes recursos:

- Regra composta: foi adicionado suporte para regras compostas de criação de jogos para jogos de 40 ou menos jogadores. Agora é possível usar declarações lógicas para criar uma regra composta para formar um jogo. Sem uma regra composta em seu conjunto de regras, para formar um jogo, todas as regras do conjunto de regras devem ser verdadeiras. Com regras compostas, você pode escolher quais regras aplicar usando os seguintes operadores lógicos: `and`, `or`, `not` e `xor`.
- Seleção flexível de equipes: expressões de propriedades de criação de jogos atualizadas para apoiar a seleção de um subconjunto de todas as equipes disponíveis.
- Listas de sequências mais longas: Aumentamos o número máximo de sequências de caracteres de 10 para 100 em uma lista de sequências de valores de atributos do jogador.

Saiba mais:

- [Amazon GameLift ServersFlexMatchguia do desenvolvedor](#):
 - [FlexMatchtipos de regras](#)
 - [FlexMatchexpressões de propriedade](#)
- [AttributeValue: SL](#)

28 de outubro de 2021: Amazon GameLift Servers adiciona suporte para frotas multirregionais na região Ásia-Pacífico (Osaka); Amazon GameLift ServersFleetIQ adiciona suporte para processadores Graviton2 AWS

Versões atualizadas do SDK: AWS [SDK 1.9.133](#)

O Amazon GameLift Servers agora está disponível na região Ásia-Pacífico (Osaka). Agora, os desenvolvedores de jogos podem implantar instâncias em Osaka usando uma frota GameLift multirregional.

Agora é possível usar servidores de jogos hospedados no Graviton2, com base na arquitetura de processador baseado em ARM, para obter maior desempenho a um custo menor quando comparado às opções de computação equivalentes baseadas em Intel.

Destaques:

- O Amazon GameLift Servers agora está disponível na região Ásia-Pacífico (Osaka).
- Amazon GameLift Servers FleetIQ Agora, os grupos de servidores de jogos podem ser configurados para gerenciar as famílias de instâncias do Graviton2 c6g, m6g e r6g.

Saiba mais:

- [Amazon GameLift Servers Frota multirregional](#)
- [CreateGameServerGroup](#)
- [AWS processador de grávitons](#)

20 de setembro de 2021: Amazon GameLift Servers lança plugin para Unity

O Amazon GameLift Servers plug-in para Unity versão 1.0.0 contém bibliotecas e interface de usuário nativa que facilitam o acesso aos Amazon GameLift Servers recursos e a Amazon GameLift Servers integração ao seu jogo Unity. Você pode usar o Amazon GameLift Servers plug-in do Unity para acessar Amazon GameLift Servers APIs e implantar AWS CloudFormation modelos para cenários de jogos comuns. O plug-in também inclui um jogo de amostra que funciona com os cenários de amostra. Você pode usar o Amazon GameLift Servers Local para ver as mensagens passadas entre o cliente do jogo e o servidor do jogo para saber como um jogo típico interage com Amazon GameLift Servers.

O plug-in para Unity é compatível com o Unity 2019.4 LTS e 2020.3 LTS.

Destaques:

- Crie, execute e modifique um jogo de amostra com cenários diferentes ou crie o seu próprio.
- Implante exemplos de AWS CloudFormation cenários para cenários de jogo típicos, incluindo somente autenticação, frota de região única, frotas multirregionais com fila e matchmaker personalizado, Spot Fleets com fila e matchmaker personalizado, e. FlexMatch

Saiba mais:

- [Integrando jogos com o Amazon GameLift Servers plugin para Unity](#)

30 de junho de 2021: FlexMatch adiciona a regra BatchDistance

É possível usar o tipo de regra BatchDistance para especificar uma sequência de caracteres ou um atributo numérico, trazendo uma série de benefícios para cada segmento.

Destaques:

- Para jogos grandes (>40 jogadores), em vez de equilibrar uniformemente os jogadores apenas por habilidade, agora é possível obter o mesmo equilíbrio com base na habilidade, nos modos e nos mapas. Verifique se todos no jogo estejam em uma faixa de habilidade, agrupe vários atributos numéricos, como liga ou estilo de jogo, e agrupe de acordo com atributos de sequência, como mapa ou modo de jogo. Também é possível criar expansões ao longo do tempo. Por exemplo, você pode criar uma expansão para permitir que uma faixa maior de nível de habilidade entre no jogo quanto mais tempo o jogador estiver esperando.

Para jogos com menos de 40 jogadores, é possível usar uma nova expressão de regras simplificada.

3 de junho de 2021: atualizações em Amazon GameLift Servers tempo real do SDK do cliente e do SDK do servidor

Versões atualizadas do SDK: SDK 1.2.0 de cliente em tempo real, SDK 3.4.0 de servidor para Unreal

Com esta atualização mais recente do SDK, agora você pode integrar o IL2 CPP em seus aplicativos móveis que usam o RTS Client SDK e seguir as melhores práticas com estruturas. Agora você também pode criar o SDK Amazon GameLift Servers do servidor para Unreal versão 4.26. Essa atualização contém componentes que se integram ao seu servidor de jogos Windows ou Linux, incluindo versões C++ e C# do Amazon GameLift Servers Server SDK, Amazon GameLift Servers Local e um plug-in Unreal Engine.

Destaques:

- Foi adicionado suporte para IL2 CPP no RTS Client SDK e para criar bibliotecas nativas como estruturas, para que você possa criar clientes RTS para os dispositivos móveis mais recentes.
- É possível usar [DescribePlayerSessions\(\)](#) para obter informações de uma única sessão de jogador, para todas as sessões de jogador em uma sessão de jogo, ou para todas as sessões de jogador associadas a um único ID de jogador.
- É possível usar [GetInstanceCertificate\(\)](#) para recuperar o local do arquivo de um certificado TLS codificado em PEM associado à frota e às suas instâncias.

- Foi criado o suporte do SDK de servidor para Unreal versão 4.26.
- O SDK C# existente, versão 4.0.2, foi verificado como compatível com o Unity 2020.3. Nenhuma atualização do SDK é necessária.

Saiba mais:

- [Amazon GameLift Servers Guia do desenvolvedor](#):
 - [DescribePlayerSessions\(\)](#)
 - [GetInstanceCertificate\(\)](#)

23 de março de 2021: Amazon GameLift Servers adiciona notificações ao posicionamento da sessão de jogo

Versões atualizadas do SDK: AWS [SDK 1.8.168](#)

Agora é possível usar eventos para monitorar a atividade de posicionamento da sessão de jogo em uma fila de sessões de jogo. Crie um tópico do Amazon Simple Notification Service (Amazon SNS) para publicar notificações de eventos ou configure o rastreamento de eventos usando Eventos. CloudWatch

Destaques:

- Para cada fila, é possível definir uma sequência de texto personalizada para ser incluída em todas as mensagens do evento.
- Ao usar um tópico do Amazon SNS, é possível definir condições de acesso adicionais que limitam a publicação em filas específicas.

Saiba mais:

- Amazon GameLift Servers Guia do desenvolvedor:
 - [Configurar notificações de eventos para o posicionamento de sessões de jogo](#) (novo)
 - [Eventos de posicionamento de sessões de jogo](#) (novo)
- [Referência da API \(SDK da AWS\)](#)
 - Novos parâmetros da fila de sessões de jogo NotificationTarget e CustomEventData: [GameSessionQueue](#), [CreateGameSessionQueueUpdateGameSessionQueue](#)
- [Fórum da Amazon GameLift Servers](#)

16 de março de 2021: Amazon GameLift Servers adiciona frotas multirregionais, seis novas regiões

Versões atualizadas do SDK: AWS [SDK 1.8.163](#)

Amazon GameLift ServersA hospedagem gerenciada agora está disponível em 21 AWS regiões. As novas regiões são Cidade do Cabo (af-south-1), Bahrein (me-south-1), Hong Kong (ap-east-1), Milão (eu-south-1), Paris (eu-west-3) e Estocolmo (eu-north-1).

Com o novo recurso de frotas em Amazon GameLift Servers vários locais, agora você pode configurar uma única frota para hospedar seus servidores de jogos em qualquer uma ou em todas as 20 regiões Amazon GameLift Servers suportadas (exceto a região de Pequim). Esse recurso visa reduzir significativamente o trabalho necessário para configurar e manter os recursos de Amazon GameLift Servers hospedagem globalmente. É possível criar frotas com vários locais nas seguintes AWS regiões: us-east-1 (Norte da Virgínia), us-west-2 (Oregon), eu-central-1 (Frankfurt), (eu-west-1Irlanda), ap-southeast-2 (Sydney), (Tóquio) e ap-northeast-1 ap-northeast-2 (Seul). Em todas as outras regiões, é possível continuar configurando frotas de localização única conforme necessário. Todas as frotas criadas antes dessa versão são frotas de localização única. O uso de frotas com vários locais não afeta seus custos de hospedagem. Amazon GameLift Servers preço é baseado no tipo, localização e volume de instâncias que você usa. (Para obter mais informações, consulte [Amazon GameLift Servers preços](#).) AWS CloudFormation o suporte para frotas em vários locais estará disponível em breve.

Note

As frotas com vários locais não estão disponíveis nas regiões da China. Amazon GameLift Serversrecursos que residem nas regiões da China não podem interagir ou ser usados por recursos em outras Amazon GameLift Servers regiões.

Destaques:

- Com uma frota com vários locais, adicione explicitamente uma lista de locais remotos. Amazon GameLift Serversimplanta instâncias do mesmo tipo e configuração, incluindo a configuração de compilação e tempo de execução, na região de origem da frota e em todos os locais adicionados.
- Ajuste as configurações de capacidade e o dimensionamento para cada local de forma independente. As políticas de ajuste de escala automático se aplicam a uma frota inteira, mas é possível ativá-las ou desativá-las por local.

- Inicie novas sessões de jogo em locais específicos da frota. Ao usar filas de sessões de jogo ou criação de jogos para organizar sessões de jogo, agora é possível priorizar onde as novas sessões de jogo começam por local, custo de hospedagem e latência do jogador.
- Obtenha métricas de hospedagem no Amazon GameLift Servers console, agregadas para todos os locais em uma frota ou divididas por localização de cada frota.

Saiba mais:

- [Blog de tecnologia de jogos da Amazon](#)
- [Referência da API \(SDK da AWS\)](#)
 - Novas operações de localização da frota: [CreateFleetLocations](#), [DescribeFleetLocationAttributes](#), [DescribeFleetLocationCapacity](#), [DescribeFleetLocationUtilization](#), [DeleteFleetLocations](#)
 - Operações de frota atualizadas, com novo suporte em vários locais: [CreateFleet](#), [DescribeFleetCapacityEC2InstanceLimits](#), [DescribeInstances](#), [StopFleetActions](#), [StartFleetActions](#)
 - Operações de posicionamento de sessões de jogo atualizadas, com nova prioridade e capacidade de filtragem: [CreateGameSessionQueue](#), [DescribeGameSessionQueues](#), [UpdateGameSessionQueue](#)
 - Operações de criação de sessão de jogo atualizadas, com novo suporte de localização: [CreateGameSession](#), [DescribeGameSessions](#), [DescribeGameSessionDetails](#), [SearchGameSessions](#)
- [Amazon GameLift Servers Guia do desenvolvedor](#):
 - [Amazon GameLift Servers locais de serviço](#) (atualizado)
 - [Personalize seu Amazon GameLift Servers EC2 frotas gerenciadas](#) (novo)
 - [Escalando a capacidade de hospedagem de jogos com Amazon GameLift Servers](#) (atualizado)
 - [Personalize uma fila de sessões de jogo](#) (novo)
 - [Detalhes da frota no Amazon GameLift Servers console](#) (atualizado)
- [Fórum da Amazon GameLift Servers](#)

9 de fevereiro de 2021: Amazon GameLift Servers estende o suporte para instâncias AMD, autônomas FlexMatch

Versões atualizadas do SDK: AWS [SDK 1.8.139](#)

Esta versão inclui as seguintes alterações:

- Amazon GameLift ServersFleetIQAgora, os grupos de servidores de jogos podem ser configurados para gerenciar as famílias de instâncias AMD C5a, M5a e R5a. Os tipos de EC2 instância da Amazon compatíveis, conforme listados para o GameServerGroup [InstanceDefinition](#), agora incluem o seguinte:
 - 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

Observação: no momento, as instâncias da AMD não FleetIQ estão disponíveis para uso na AWS região da China (Pequim). Para obter as informações mais recentes, consulte [Disponibilidade de atributos e diferenças de implementação](#).

- Amazon GameLift ServersA hospedagem gerenciada de jogos agora oferece suporte a instâncias AMD na região da China (Pequim), operadas pela Sinnet. As novas famílias de instâncias AMD incluem M5a e R5a. Os tipos de EC2 instância compatíveis, conforme listados para frota [InstanceType](#), agora incluem o seguinte:
 - 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 ServersFlexMatchagora pode ser usado como uma solução autônoma de matchmaking na região da China (Pequim), operada pela Sinnet. Os clientes podem criar um FlexMatch matchmaker na região de Pequim e configurar o [FlexMatchMode](#) parâmetro como STANDALONE. Para obter mais informações sobre FlexMatch, seja com hospedagem Amazon GameLift Servers gerenciada ou com uma solução sem Amazon GameLift Servers hospedagem, consulte o [Guia do Amazon GameLift Servers FlexMatch desenvolvedor](#).
- Ao configurar notificações de eventos para Amazon GameLift ServersFlexMatch, agora você pode designar um tópico FIFO do Amazon SNS como alvo da notificação. Para obter mais informações, consulte:
 - [MatchmakingConfiguration NotificationTarget](#), Referência Amazon GameLift Servers da API

- [Configurar notificação de FlexMatch eventos](#), Guia Amazon GameLift Servers FlexMatch do desenvolvedor
- [Apresentando o Amazon SNS FIFO — First-in-first-out Pub/Sub mensagens](#), blog de notícias AWS

22 de dezembro de 2020: o SDK Amazon GameLift Servers do servidor é compatível com Unreal Engine 4.25 e Unity 2020

Versões atualizadas do SDK: Amazon GameLift Servers Server SDK 4.0.2, versão 3.3.3 do plugin Unreal

A versão mais recente do SDK Amazon GameLift Servers do servidor contém os seguintes componentes:

- O plug-in Unreal atualizado foi atualizado para compatibilidade com o Unreal Engine 4.25. A API não foi alterada.
- O SDK C# existente, versão 4.0.2, foi verificado como compatível com o Unity 2020. Nenhuma atualização do SDK é necessária.

Faça o download da versão mais recente do SDK Amazon GameLift Servers do servidor ao [Amazon GameLift Servers](#)começar.

24 de novembro de 2020: Amazon GameLift ServersFlexMatch agora disponível para jogos hospedados em qualquer lugar

Versões atualizadas do SDK: AWS [SDK 1.8.95](#)

Amazon GameLift ServersFlexMatch é um serviço personalizável de matchmaking para jogos multijogador. Projetado inicialmente para usuários de hospedagem Amazon GameLift Servers gerenciada, agora FlexMatch pode ser integrado a jogos que usam outros sistemas de hospedagem peer-to-peer, incluindo computação local proprietária e tipos primitivos de computação em nuvem. Os jogos usados Amazon GameLift Servers FleetIQ para hospedagem de jogos na Amazon agora EC2 podem implementar matchmaking com FlexMatch.

FlexMatch fornece um algoritmo robusto de matchmaking e uma linguagem de regras que oferece ampla latitude para personalizar o processo de matchmaking para que os jogadores sejam combinados com base nas principais características do jogador e na latência relatada. Além disso, FlexMatch oferece um fluxo de trabalho de solicitação de matchmaking que suporta recursos

como grupos de jogadores, aceitação de jogadores e preenchimento de partidas. Quando você usa FlexMatch com hospedagem Amazon GameLift Servers gerenciada ou Amazon GameLift Servers Realtime, o matchmaker usa automaticamente Amazon GameLift Servers para encontrar recursos de hospedagem e iniciar uma nova sessão de jogo para partidas recém-formadas. Ao usar FlexMatch como um serviço independente, o matchmaker devolve os resultados da partida ao seu jogo, que pode então iniciar uma nova sessão de jogo usando sua solução de hospedagem.

As operações de API para FlexMatch fazem parte da API de Amazon GameLift Servers serviço, que está incluída no AWS SDK e no AWS Command Line Interface (AWS CLI). Essa versão inclui essas atualizações para oferecer suporte à criação de jogos independente:

- O recurso `MatchmakingConfiguration` da API tem as seguintes alterações:
 - Nova propriedade, `FlexMatchMode` indica se o matchmaker está sendo usado com hospedagem Amazon GameLift Servers gerenciada ou como matchmaking independente.
 - A propriedade `GameSessionQueueArns` não é necessária quando o `FlexMatchMode` é definido como independente.
 - Essas propriedades não são usadas com criação de jogos independente:
`AdditionalPlayerCount`, `BackfillMode`, `GameProperties`, `GameSessionData`.
- O atributo de preenchimento automático não está disponível com criação de jogos independente.

24 de novembro de 2020: instâncias AMD agora disponíveis em Amazon GameLift Servers

Versões atualizadas do SDK: AWS [SDK 1.8.95](#)

A lista de tipos de EC2 instância da Amazon compatíveis Amazon GameLift Servers agora inclui três novas famílias de instâncias: C5a, M5a e R5a. Essas famílias consistem em instâncias AMD otimizadas para computação que são alimentadas por processadores AMD EPYC executados em frequências de até 3,3 GHz. As instâncias da AMD são compatíveis com x86; os jogos em execução no momento Amazon GameLift Servers podem ser implantados nos tipos de instância da AMD sem alterações. As novas instâncias estão disponíveis nas seguintes AWS regiões: Leste dos EUA (Norte da Virgínia e Ohio), Oeste dos EUA (Oregon e Norte da Califórnia), Canadá Central (Montreal), América do Sul (São Paulo), UE Central (Frankfurt), Oeste da UE (Londres e Irlanda), Ásia-Pacífico Sul (Mumbai), Nordeste da Ásia-Pacífico (Seul e Tóquio) e Sudeste da Ásia-Pacífico (Cingapura e Sydney).

As novas instâncias AMD incluem:

- 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

Saiba mais:

- [Blog de tecnologia de jogos da Amazon](#)
- [Amazon GameLift Servers preços de instâncias](#)
- [EC2 Instâncias da Amazon com processadores AMD EPYC](#)
- [Fórum da Amazon GameLift Servers](#)

11 de novembro de 2020: atualização da versão do SDK Amazon GameLift Servers do servidor

Versões atualizadas do SDK: Amazon GameLift Servers Server SDK 4.0.2

A nova versão 4.0.2 do SDK de servidor corrige um problema conhecido com a operação `StartMatchBackfill()` da API. Essa operação agora retorna uma resposta correta para uma solicitação de preenchimento de jogos.

O problema não afetou o processo de preenchimento de jogos e não há alteração na forma como esse recurso funciona. O problema pode ter afetado as mensagens de log e o tratamento de erros nas solicitações de preenchimento de jogos.

Faça o download da versão mais recente do SDK Amazon GameLift Servers do servidor ao [Amazon GameLift Servers começar](#).

5 de novembro de 2020: novas personalizações de FlexMatch algoritmos

FlexMatchos usuários agora podem ajustar os seguintes comportamentos padrão para o processo de matchmaking. Essas personalizações são definidas em um conjunto de regras de criação de jogos. Não há alterações no Amazon GameLift Servers SDKs.

- Priorize os tickets de preenchimento: é possível escolher entre aumentar ou diminuir a forma como os tickets de preenchimento de jogos são priorizados ao pesquisar jogos aceitáveis. Priorizar os

tickets de preenchimento é útil quando o recurso de preenchimento automático está ativado. Use a propriedade `backfillPriority` do algoritmo.

- Pré-classifique para otimizar a consistência e a eficiência do jogo: configure seu matchmaker para pré-classificar o pool de tickets antes de agrupar os tickets para avaliação. Ao pré-classificar os tickets com base nos principais atributos dos jogadores, os jogos resultantes tendem a ter jogadores mais semelhantes nesses atributos. Também é possível aumentar a eficiência no processo de avaliação pré-classificando os mesmos atributos usados nas regras de jogo. Use a propriedade `sortByAttributes` do algoritmo com a propriedade `strategy` definida como “classificada”.
- Ajuste como os tempos de espera da expansão são acionados: escolha entre ativar expansões com base na idade do ticket mais novo (padrão) ou do mais antigo em um jogo incompleto. O acionamento do ticket mais antigo tende a concluir os jogos mais rapidamente, enquanto o acionamento do ticket mais novo leva a uma maior qualidade da jogo. Use a propriedade `expansionAgeSelection` do algoritmo.

17 de setembro de 2020: Amazon GameLift Servers atualiza o SDK do servidor

Versões atualizadas do SDK: Amazon GameLift Servers Server SDK 4.0.1

O novo SDK de servidor contém as seguintes atualizações:

- Versão 4.0.1 da API C#
 - A operação [TerminateGameSession\(\)](#) da API não é mais compatível. Substitua por uma chamada para [ProcessEnding\(\)](#) para encerrar a sessão do jogo e o processo do servidor.
 - Um problema conhecido com a operação [GetInstanceCertificate\(\)](#) foi corrigido.
 - A operação [GetTerminationTime\(\)](#) agora retorna um valor do tipo de dados `AwsDateTimeOutcome`.
- Versão 3.4.1 da API C#
 - A operação [TerminateGameSession\(\)](#) não é mais compatível. Substitua por uma chamada para [ProcessEnding\(\)](#) para encerrar a sessão do jogo e o processo do servidor.
- Plug-in do Unreal Engine versão 3.3.2
 - A operação [TerminateGameSession\(\)](#) não é mais compatível. Substitua por uma chamada para [ProcessEnding\(\)](#) para encerrar a sessão do jogo e o processo do servidor.
 - A operação `OnUpdateGameSession` de retorno de chamada é adicionada a [FProcessParâmetros](#) para oferecer suporte ao preenchimento de jogos.

Faça o download da versão mais recente do SDK Amazon GameLift Servers do servidor ao [Amazon GameLift Servers](#) começar.

27 de agosto de 2020: Amazon GameLift Servers FleetIQ para hospedagem de jogos com a Amazon EC2 (disponibilidade geral)

Versões atualizadas do SDK: AWS [SDK 1.8.36](#)

A Amazon GameLift Servers FleetIQ solução para hospedagem de jogos de baixo custo e baseada em nuvem na Amazon agora EC2 está disponível ao público em geral. Amazon GameLift Servers FleetIQ oferece aos desenvolvedores a capacidade de hospedar servidores de jogos diretamente nas Instâncias EC2 Spot da Amazon, otimizando sua viabilidade para hospedagem de jogos. Os desenvolvedores de jogos podem usar Amazon GameLift Servers FleetIQ com novos jogos ou para complementar a capacidade dos jogos existentes. Essa solução suporta o uso de contêineres ou outros AWS serviços, como AWS Shield e Amazon Elastic Container Service (Amazon ECS).

Essa versão de disponibilidade geral inclui as seguintes atualizações da Amazon GameLift Servers FleetIQ solução:

- A nova operação de API `DescribeGameServerInstances` retorna informações, incluindo status, sobre todas as instâncias ativas de um grupo de servidores de Amazon GameLift Servers FleetIQ jogos.
- A nova estratégia de balanceamento, `ON_DEMAND_ONLY`, configura um grupo de servidores de jogos para usar somente instâncias sob demanda. É possível atualizar a estratégia de balanceamento de um grupo de servidores de jogos a qualquer momento, possibilitando alternância entre o uso de instâncias spot e instâncias sob demanda, conforme necessário.
- Os seguintes elementos de pré-visualização foram descartados para disponibilidade geral:
 - Uso de chaves de classificação personalizadas para recursos do servidor de jogos. Os servidores de jogos podem ser classificados com base no carimbo de data e hora do registro.
 - Marcação de recursos do servidor de jogos.

16 de abril de 2020: Amazon GameLift Servers atualiza o SDK do servidor para Unity e Unreal Engine

Versões atualizadas do SDK: Amazon GameLift Servers Server SDK 4.0.0, Local 1.0.5 Amazon GameLift Servers

A versão mais recente do SDK de servidor do Amazon GameLift Servers contém os seguintes componentes atualizados:

- SDK C# versão 4.0.0 atualizado para Unity 2019
- Plug-in do Unreal versão 3.3.1 atualizado para Unreal Engine versões 4.22, 4.23 e 4.24.
- Amazon GameLift Servers Versão local 1.0.5 atualizada para testar integrações que usam o SDK do servidor C# versão 4.0.0.

Faça o download da versão mais recente do SDK Amazon GameLift Servers do servidor ao [Amazon GameLift Servers](#) [começar](#).

2 de abril de 2020: Amazon GameLift Servers FleetIQ disponível para hospedagem de jogos em EC2 (prévia pública)

Versões atualizadas do SDK: AWS [SDK 1.7.310](#)

O recurso FleetIQ do Amazon GameLift Servers otimiza a viabilidade de instâncias spot de baixo custo para uso com hospedagem de jogos. Esse recurso agora é estendido para clientes que desejam gerenciar seus recursos de hospedagem diretamente, em vez de por meio do serviço gerenciado do Amazon GameLift Servers. Essa solução suporta o uso de contêineres ou outros AWS serviços, como AWS Shield e Amazon Elastic Container Service (Amazon ECS).

Saiba mais:

[GameTech postagem no blog](#) sobre Amazon GameLift Servers FleetIQ

19 de dezembro de 2019: gerenciamento aprimorado AWS de Amazon GameLift Servers recursos

Versões atualizadas do SDK: AWS [SDK 1.7.249](#)

Agora você pode aproveitar as ferramentas de gerenciamento de AWS Amazon GameLift Servers recursos com recursos. Em particular, todos os principais Amazon GameLift Servers recursos — builds, scripts, frotas, filas de sessões de jogo, configurações de matchmaking e conjuntos de regras de matchmaking — agora recebem valores de Amazon Resource Name (ARN). O ARN de um recurso fornece um identificador consistente que é exclusivo em todas as AWS regiões. Eles podem ser usados para criar políticas de permissões específicas de recursos AWS Identity and Access Management (IAM). Agora os recursos recebem um ARN e o identificador de recurso pré-existente, que não é específico para a região.

Além disso, os recursos do Amazon GameLift Servers agora oferecem suporte à marcação. Você pode usar tags para organizar recursos, criar políticas de permissões do IAM para gerenciar o acesso a grupos de recursos, personalizar detalhamentos de AWS custos etc. Ao gerenciar tags para recursos do Amazon GameLift Servers, use as ações `TagResource()`, `UntagResource()` e `ListTagsForResource()` da API do Amazon GameLift Servers.

Saiba mais:

- [TagResource](#) na Referência de API do Amazon GameLift Servers
- [Marcar recursos da AWS](#) na Referência geral da AWS
- [Nomes de recursos da Amazon](#) na Referência geral da AWS

14 de novembro de 2019: Novos modelos do AWS CloudFormation , atualizações na região da China (Pequim)

Versões atualizadas do SDK: AWS [SDK 1.7.210](#)

AWS CloudFormation modelos para Amazon GameLift Servers

Amazon GameLift Servers recursos agora podem ser criados e gerenciados por meio do AWS CloudFormation. Os modelos existentes de AWS CloudFormation construção e frota foram atualizados para se alinharem aos recursos atuais, e novos modelos agora estão disponíveis para scripts, filas, configurações de matchmaking e conjuntos de regras de matchmaking. AWS CloudFormation os modelos simplificam muito a tarefa de gerenciar grupos de AWS recursos relacionados, principalmente ao implantar jogos em várias regiões.

Saiba mais:

- [Amazon GameLift Servers referência do tipo de recurso](#) no Guia AWS CloudFormation do usuário
- [Gerenciamento Amazon GameLift Servers recursos de hospedagem usando AWS CloudFormation](#) no Guia do Amazon GameLift Servers desenvolvedor

As traduções são geradas por tradução automática. Em caso de conflito entre o conteúdo da tradução e da versão original em inglês, a versão em inglês prevalecerá.