



Panduan Hosting

Amazon GameLift Servers



Amazon GameLift Servers: Panduan Hosting

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

Merek dagang dan tampilan dagang Amazon tidak boleh digunakan sehubungan dengan produk atau layanan apa pun yang bukan milik Amazon, dengan cara apa pun yang dapat menyebabkan kebingungan di antara pelanggan, atau dengan cara apa pun yang merendahkan atau mendiskreditkan Amazon. Semua merek dagang lain yang tidak dimiliki oleh Amazon merupakan properti dari masing-masing pemilik, yang mungkin berafiliasi, terkait dengan, atau disponsori oleh Amazon, atau tidak.

Table of Contents

Apa itu Amazon GameLift Servers?	1
Apa yang dapat Anda lakukan dengan Amazon GameLift Servers	1
Cara bekerja dengan Amazon GameLift Servers	2
Amazon GameLift Servers solusi	2
Opsi hosting	3
FlexMatch untuk perijinan	7
FleetIQ untuk hosting Amazon EC2 yang dikelola sendiri	7
Realtime server dengan logika server yang dapat disesuaikan	8
Arsitektur solusi hosting terkelola	9
Komponen game dengan hosting	9
Sumber daya solusi hosting	11
Cara kerja Amazon GameLift Servers	12
Server hosting game	13
Menjalankan sesi game	13
Penskalaan kapasitas armada	15
Memantau Amazon GameLift Servers	15
Menggunakan AWS sumber daya lain	16
Cara kerja kontainer	16
Komponen armada kontainer	17
Arsitektur umum	19
Fitur inti	20
Bagaimana pemain terhubung ke game	27
Lokasi layanan	28
AWS Lokasi yang didukung	28
Lokasi untuk hosting terkelola	33
Lokasi untuk Amazon GameLift Servers Di Mana Saja	34
Lokasi untuk Amazon GameLift Servers FlexMatch	35
Amazon GameLift Servers di Tiongkok	35
Harga	36
Hasilkan perkiraan harga	36
Kelola biaya hosting game	40
Memulai	42
Sebelum Anda mulai	42
Opsi orientasi cepat	42

Opsi pengembangan khusus	43
Amazon GameLift Servers contoh	43
Contoh server game kustom	44
Amazon GameLift ServersRealtimecontoh	44
Mengatur sebuah Akun AWS	45
Mendaftar untuk Akun AWS	45
Buat pengguna dengan akses administratif	46
Tetapkan izin pengguna untuk Amazon GameLift Servers	47
Menyiapkan akses terprogram untuk pengguna	48
Siapkan akses terprogram untuk game Anda	49
Contoh izin IAM	50
Menyiapkan peran layanan IAM	54
Dapatkan alat pengembangan	59
Untuk server game	59
Untuk layanan klien game	62
Untuk manajemen sumber daya	62
Untuk Amazon GameLift ServersRealtime	63
Tutorial: Orientasi cepat dengan pembungkusnya Amazon GameLift Servers	64
Prasyarat	65
Gambaran Umum	65
Langkah 1: Dapatkan dan bangun pembungkus server game	65
Langkah 2: Siapkan build server game Anda	66
Langkah 3: Konfigurasi pembungkus untuk armada Anda	67
Langkah 4: Unggah build server game	68
Langkah 5: Buat EC2 armada yang dikelola	69
Langkah 6: Buat dan sambungkan ke sesi permainan	70
Langkah 7: Kelola dan pantau armada Anda	71
Langkah 8: Skala server game Anda	72
Memecahkan masalah umum	73
Langkah selanjutnya	75
Jelajahi dengan plugin mesin game	75
Alur kerja plugin	76
Plugin untuk Unreal Engine	77
Plugin untuk Unity (server SDK 5.x)	110
Plugin untuk Unity (server SDK 4.x)	132
Peta jalan EC2 pengembangan terkelola	159

Peta jalan pengembangan kontainer terkelola	166
Peta jalan pengembangan di mana saja	173
Peta jalan pengembangan hosting hibrida	179
Mempersiapkan game untuk Amazon GameLift Servers	188
Integrasikan game dengan server game khusus	188
Amazon GameLift Servers interaksi	189
Mengintegrasikan server game	194
Integrasikan klien game	205
Mesin permainan dan Amazon GameLift Servers	212
Rancang layanan backend Anda	237
Mengautentikasi pemain Anda	237
Backend tanpa server	237
WebSocketberbasis backend	239
Siapkan untuk pengembangan berulang	241
Membangun lingkungan pengujian berbasis cloud	243
Siapkan pengujian lokal	247
Bekerja dengan Agen	252
Siapkan pengujian lokal (warisan)	254
Penyiapan Amazon GameLift Servers lokal	256
Uji server game	256
Uji server game dan klien	259
Variasi dengan lokal	262
Menambahkan FlexMatch perjodohan	262
Dapatkan data armada	263
Mengintegrasikan game dengan Amazon GameLift Servers Realtime	264
Apa itu Realtime server?	264
Mengelola sesi permainan	265
Interaksi server klien	265
Menyesuaikan server	266
Menyebarkan dan memperbarui	267
Mengintegrasikan klien game	267
Menyesuaikan skrip Realtime	273
Menyebarkan perangkat lunak server game	280
Menerapkan build server khusus	280
Menyebarkan untuk hosting terkelola	281
Menyebarkan untuk hosting di mana saja	281

Membuat paket file bangunan game Anda	281
Menambahkan skrip instalasi bangunan	282
Buat build untuk hosting terkelola	285
Perbarui build server game untuk hosting terkelola	289
Membangun gambar kontainer	291
Buat gambar kontainer server game	292
Dorong gambar kontainer ke Amazon ECR	294
Menyebarkan skrip Amazon GameLift Servers Realtime	295
Buat paket file skrip	296
Unggah file skrip dari direktori lokal	296
Unggah file skrip dari Amazon S3	297
Memperbarui file skrip	299
Menyiapkan armada hosting	301
Karakteristik armada	301
Cara kerja pembuatan armada	302
Armada yang EC2 dikelola	302
Alur kerja pembuatan EC2 armada terkelola	303
Buat EC2 armada terkelola	304
Sesuaikan armada EC2 terkelola Anda	314
Perbarui konfigurasi armada	320
Debug masalah armada	324
Terhubung dari jarak jauh ke instance armada	328
Peering VPC	336
Armada kontainer yang dikelola	343
Buat armada kontainer	344
Perbarui armada kontainer	348
Sesuaikan armada kontainer	350
Buat definisi grup kontainer	358
Memperbarui definisi grup kontainer	363
Hapus definisi grup kontainer	367
Armada di mana saja	368
Alur kerja pembuatan armada di mana saja	369
Buat armada Anywhere	369
Abstrak armada dengan alias	379
Membuat alias	380
Mengedit alias	381

Mengelola penempatan sesi permainan dengan antrian	383
Karakteristik antrian	383
Membuat antrean	384
Sesuaikan antrian	388
Praktik terbaik	389
Tentukan ruang lingkup antrian	390
Membangun antrian multi-lokasi	391
Prioritaskan penempatan sesi permainan	393
Evaluasi metrik antrean	400
Desain untuk Instans Spot	402
Mengatur notifikasi kejadian	404
Menyiapkan topik SNS	405
Siapkan topik SNS dengan enkripsi sisi server	406
Mengatur EventBridge	407
Tutorial: Buat antrian dengan Instans Spot	408
Langkah 1: Tentukan ruang lingkup antrean Anda	409
Langkah 2: Buat Infrastruktur armada Spot	409
Langkah 3: Tetapkan alias untuk setiap armada	410
Langkah 4: Buat antrean dengan tujuan	411
Langkah 5: Tambahkan batas latensi pada antrean	413
Ringkasan	415
Penskalaan kapasitas hosting	417
Untuk mengelola kapasitas armada di konsol	418
Mengatur batas kapasitas	418
Mengatur batas kapasitas	419
Mengatur kapasitas armada secara manual	420
Tangguhkan penskalaan otomatis	420
Mengatur kapasitas armada secara manual	421
Kapasitas armada skala otomatis	422
Penskalaan otomatis berbasis target	423
Penskalaan otomatis berbasis aturan	424
Armada kontainer skala	429
Mempersiapkan peluncuran game	432
Bersiap-siap	432
Bersiaplah untuk pengujian	433
Bersiaplah untuk peluncuran	433

Rencanakan untuk pasca-peluncuran	434
Mengelola sumber daya hosting dengan AWS CloudFormation	435
Praktik terbaik	435
Menggunakan AWS CloudFormation tumpukan	436
Tumpukan untuk satu lokasi	436
Tumpukan untuk beberapa wilayah	438
Memperbarui build	441
Terapkan pembaruan build secara otomatis	442
Terapkan pembaruan build secara manual	442
Bagaimana rollback bekerja	443
Pemantauan Amazon GameLift Servers	444
Lacak hosting game di konsol	445
Dasbor hosting	445
Server game dibangun	447
Amazon GameLift ServersRealtimeskrip	449
Armada	450
Rincian armada	451
Sesi permainan dan sesi pemain	454
Alias	459
Antrian sesi permainan	461
Monitor dengan CloudWatch	463
Dimensi metrik	464
Metrik armada	465
Metrik antrian	477
Metrik-metrik FlexMatch	480
Metrik-metrik FleetIQ	484
Mencatat panggilan API	487
Amazon GameLift Servers informasi di CloudTrail	487
Memahami Amazon GameLift Servers entri file log	488
Mencatat pesan server	490
Logging untuk server kustom	491
Penebangan untuk Amazon GameLift Servers Realtime	493
Keamanan	499
Perlindungan data	500
Enkripsi diam	502
Enkripsi bergerak	502

Privasi lalu lintas jaringan internet	502
Manajemen identitas dan akses	503
Audiens	503
Mengautentikasi dengan identitas	504
Mengelola akses menggunakan kebijakan	507
Bagaimana Amazon GameLift Servers bekerja dengan IAM	510
Contoh kebijakan berbasis identitas	518
Pemecahan Masalah	523
AWS kebijakan terkelola	525
Pencatatan dan pemantauan dengan Amazon GameLift Servers	527
Validasi kepatuhan	528
Ketahanan	529
Keamanan infrastruktur	530
Konfigurasi dan analisis kerentanan	531
Praktik terbaik keamanan	532
Jangan membuka port ke Internet	532
Pelajari selengkapnya	533
Panduan referensi	534
API Layanan (AWS SDK)	534
Kelola sumber daya Amazon GameLift Servers hosting	534
Memulai sesi game dan bergabung dengan pemain	538
Server SDK 5.x	540
Pembaruan Server SDK 5.x	540
Migrasi ke server SDK 5.x	542
Server C++ SDK	543
C# server SDK	594
Pergi SDK server	637
SDK server C ++ (Tidak nyata): Tindakan	664
Server SDK 4 dan sebelumnya	710
Server SDK for C++: Tindakan	711
Server SDK untuk C #: Tindakan	735
Server SDK untuk Unreal: Tindakan	757
Amazon GameLift ServersRealtimereferensi	773
Referensi API klien waktu nyata (C #)	773
Realtimereferensi skrip	787
Acara penempatan sesi permainan	795

Sintaksis peristiwa penempatan	795
PlacementFulfilled	796
PlacementCancelled	798
PlacementTimedOut	799
PlacementFailed	800
Versi AMI	801
Titik akhir dan kuota layanan	803
Suar ping UDP	803
Cara kerja ping beacon UDP	803
Kasus penggunaan umum untuk suar ping UDP	804
Mendapatkan titik akhir suar	805
Menerapkan pengukuran latensi	806
Contoh kode	807
Catatan rilis dan versi SDK	815
Versi SDK	815
Catatan rilis	830
.....	ccccclxxvi

Apakah Amazon GameLift Servers itu?

Gunakan Amazon GameLift Servers untuk menyebarkan, mengoperasikan, dan menskalakan server khusus dan berbiaya rendah di cloud untuk game multipemain berbasis sesi. Dibangun di atas infrastruktur komputasi AWS global, Amazon GameLift Servers membantu menghadirkan server game berkinerja tinggi dan keandalan tinggi sambil secara dinamis menskalakan penggunaan sumber daya Anda untuk memenuhi permintaan pemain di seluruh dunia.

Apa yang dapat Anda lakukan dengan Amazon GameLift Servers

Amazon GameLift Servers mendukung kasus penggunaan ini dan banyak lagi:

- Host server game multiplayer kustom Anda sendiri di cloud dengan EC2 hosting Amazon GameLift Servers terkelola.
- Jalankan sumber daya hosting terkelola berbiaya rendah menggunakan [Instans Spot Amazon Elastic Compute Cloud EC2 \(Amazon\)](#).
- Host server game kontainer Anda untuk fleksibilitas di seluruh platform dan untuk mendukung migrasi dengan Amazon GameLift Servers kontainer terkelola.
- Buat solusi hosting hybrid untuk mendukung hosting lokal and/or multi-cloud sambil mengelola sesi game di satu tempat dengan Amazon GameLift Servers Anywhere.
- Buat sistem perjodohan yang kuat untuk game multipemain Anda. Amazon GameLift Servers FlexMatch
- Tingkatkan kapasitas hosting terkelola Anda secara otomatis untuk memenuhi kebutuhan game Anda berdasarkan penggunaan pemain yang sebenarnya.
- Kelola sumber daya EC2 komputasi Amazon Anda untuk bermain game di satu tempat menggunakan Amazon GameLift Servers FleetIQ.
- Buat lingkungan pengujian berulang untuk server game dan build klien Anda dengan Amazon GameLift Servers Anywhere dan. EC2
- Untuk game yang tidak memerlukan server game custom-build, siapkan solusi server ringan dengan. Amazon GameLift Servers Realtime



Tip

[Mulai sekarang dengan Amazon GameLift Servers](#)

Cara bekerja dengan Amazon GameLift Servers

Gunakan alat ini untuk bekerja dengan Amazon GameLift Servers.

AWS CLI

Gunakan AWS Command Line Interface (AWS CLI) untuk melakukan panggilan ke AWS SDK, termasuk API layanan untuk Amazon GameLift Servers. Lihat [Memulai dengan AWS CLI](#) di Panduan AWS Command Line Interface Pengguna.

Konsol Amazon GameLift Servers

Gunakan [AWS Management Console for Amazon GameLift Servers untuk](#) mengonfigurasi sumber daya, mengelola penerapan server game, dan melacak metrik kinerja dan penggunaan. Amazon GameLift Servers Konsol adalah alternatif GUI untuk mengelola sumber daya secara terprogram atau dengan AWS CLI

Amazon GameLift Servers SDKs

Ini Amazon GameLift Servers SDKs berisi perpustakaan yang diperlukan untuk membangun komunikasi antara klien game Anda, server game, dan layanan game dan Amazon GameLift Servers layanan. Untuk informasi selengkapnya, lihat [Dapatkan alat Amazon GameLift Servers pengembangan](#).

SDK klien untuk Amazon GameLift Servers Realtime

SDK klien untuk Amazon GameLift Servers Realtime memungkinkan Anda menghubungkan klien game Anda ke Realtime server yang disediakan oleh Amazon GameLift Servers, bergabung dengan sesi permainan, dan tetap sinkron dengan pemain lain. Unduh [SDK](#) dan pelajari selengkapnya tentang melakukan panggilan API dengan [API Amazon GameLift Servers Realtime klien \(C#\)](#).

Amazon GameLift Servers solusi

Amazon GameLift Servers menawarkan berbagai solusi untuk pengembang yang sedang membangun game multipemain berbasis sesi.

Solusi untuk pengembang game

- [Amazon GameLift Servers opsi hosting](#)
- [Amazon GameLift Servers FlexMatch untuk perijinan](#)

- [Amazon GameLift ServersFleetIQuntuk hosting Amazon EC2 yang dikelola sendiri](#)
- [Amazon GameLift ServersRealtimedengan logika server yang dapat disesuaikan](#)

Amazon GameLift Serversopsi hosting

Ketika bekerja dengan Amazon GameLift Servers untuk mengoperasikan server game Anda, Anda memiliki beberapa opsi untuk di mana dan bagaimana server game Anda di-host. Apakah Anda ingin menggunakan sumber daya hosting yang sudah Anda miliki atau ingin mengatur hosting berbasis cloud yang dikelola olehAmazon GameLift Servers, Anda dapat membangun pengalaman hosting yang mulus untuk pemain Anda.

[Dikelola EC2](#)

[Kontainer yang dikelola](#)

[Hosting hibrida](#)

[Hosting di mana saja](#)

Dikelola EC2

Dengan EC2 hosting Amazon GameLift Servers terkelola, Anda dapat membongkar sebagian besar pekerjaan mengelola server game Anda. Pilih sumber daya komputasi dari berbagai pilihan jenis EC2 instans Amazon. Integrasikan proyek game Anda dan biarkan Amazon GameLift Servers menangani detailnya. Untuk selengkapnya tentang hosting terkelola, lihat[Cara kerja Amazon GameLift Servers](#).

[Mulailah mengembangkan solusi Amazon GameLift Servers hosting terkelola untuk game Anda.](#)

Fitur utama

- Host game multipemain yang berjalan di sistem operasi Amazon Linux atau Windows Server.
- Berikan pengalaman gameplay latensi rendah kepada pemain Anda, di mana pun mereka berada. Terapkan server game secara global di salah satu Wilayah AWS dan Local Zones yang Amazon GameLift Servers mendukung. Untuk daftar lengkap, lihat [Amazon GameLift Serverslokasi layanan](#).
- Gunakan penempatan sesi permainan Amazon GameLift Servers cerdas sehingga pemain selalu mendapatkan pengalaman pemain terbaik yang dihosting. Anda dapat mengandalkan Amazon GameLift Servers pengambilan keputusan, atau Anda dapat menyesuaikan sekitar kriteria penempatan seperti biaya, latensi pemain, dan lokasi geografis.

- Pilih cara menskalakan sumber daya hosting Anda untuk memenuhi permintaan pemain. Kelola kapasitas secara manual atau atur penskalaan otomatis. Dengan penskalaan otomatis berbasis target, Anda dapat mempertahankan buffer kapasitas idle berukuran dinamis, yang membantu Anda mengontrol biaya sambil memastikan bahwa pemain baru dapat masuk ke game dengan menunggu minimal.
- Biarkan Amazon GameLift Servers menyebarkan dan mengelola server game berbasis cloud Anda. Amazon GameLift Servers membuat sumber daya saat Anda membutuhkannya, menginstal perangkat lunak server game Anda, dan secara otomatis memulai proses untuk menyelenggarakan sesi permainan untuk pemain. Siapkan pelacakan kesehatan khusus dan biarkan Amazon GameLift Servers mendeteksi dan menyelesaikan sumber daya yang berkinerja buruk.
- Manfaatkan kemampuan Amazon GameLift Servers pemantauan untuk menilai kinerja dan penggunaan. Anda dapat melacak metrik pada faktor-faktor seperti kinerja perangkat keras, efisiensi penempatan sesi game, dan siklus hidup proses server. Anda dapat melacak sesi permainan aktif dan sesi pemain untuk mengamati penggunaan dari waktu ke waktu. Anda juga dapat mengunduh dan menyimpan log sesi game.
- Untuk hosting produksi, otomatiskan manajemen sumber daya hosting game Anda dan penerapan menggunakan AWS CloudFormation template untuk Amazon GameLift Servers dan AWS Cloud Development Kit (AWS CDK). Manfaatkan alat dan layanan integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD) seperti AWS CodePipeline.

Kontainer yang dikelola

Amazon GameLift Servers menyediakan solusi cloud hosting lengkap untuk server game kontainer. Dengan kontainer Amazon GameLift Servers terkelola, Anda dapat memanfaatkan manfaat inti dari penggunaan kontainer, seperti portabilitas, kelincahan, dan toleransi kesalahan. Fitur-fitur berikut tersedia dengan armada kontainer terkelola.

[Mulailah mengembangkan solusi hosting Amazon GameLift Servers terkelola untuk server game kontainer Anda.](#)

Fitur utama

- Kembangkan arsitektur khusus dengan wadah ringan untuk menjalankan perangkat lunak server game Anda pada sumber Amazon GameLift Servers daya hosting berbasis Linux.
- Gunakan alat Docker untuk membuat gambar kontainer berbasis Linux. Simpan gambar untuk penerapan di repositori Amazon Elastic Container Registry (Amazon ECR).

- Memberikan pengalaman pemain latensi rendah dengan menyebarkan sumber daya armada kontainer ke Zona Lokal mana pun Wilayah AWS atau yang mendukung. Amazon GameLift Servers Lihat [Amazon GameLift Servers lokasi layanan](#).
- Kelola siklus hidup armada dengan alat untuk memodelkan versi server game dan menyebarkan pembaruan armada.
- Gunakan fitur penempatan sesi Amazon GameLift Servers permainan, termasuk antrian dan FlexMatch perjodohan, untuk menemukan pertandingan sesi permainan terbaik untuk pemain Anda.
- Uji server game dan arsitektur kontainer Anda dengan Amazon GameLift Servers layanan menggunakan armada Anywhere. Uji game Anda secara lokal atau di lingkungan pengujian berbasis cloud.
- Lacak kinerja hosting game dengan metrik kinerja khusus container. Pantau kesehatan sumber daya armada Anda menggunakan metrik perangkat keras.
- Kelola sumber daya armada kontainer menggunakan AWS CloudFormation templat untuk Amazon GameLift Servers.

Hosting hibrida

Gunakan Amazon GameLift Servers layanan ini dengan kombinasi hosting terkelola dan hosting yang Amazon GameLift Servers dikelola sendiri di mana saja. Pendekatan hybrid memungkinkan Anda membangun solusi yang Anda butuhkan saat ini sambil juga mempersiapkan di mana Anda harus berada di masa depan. Skenario umum di mana solusi hibrida masuk akal meliputi:

- Perluas solusi hosting Anda ke AWS Cloud. Lengkapi kemampuan solusi hosting yang ada (perangkat keras lokal atau hosting berbasis cloud lainnya) dengan menambahkan Amazon GameLift Servers hosting terkelola. Dengan hosting terkelola, Anda dapat meningkatkan kapasitas hosting Anda atau menambahkan kapasitas “burst” untuk meningkatkan skala dengan cepat dan hanya membayar sumber daya saat Anda membutuhkannya. Anda juga dapat memanfaatkan jejak global Amazon GameLift Servers layanan ini untuk menjangkau lebih banyak pemain di seluruh dunia dan memberikan pengalaman multipemain latensi rendah yang mereka harapkan.
- Bersiaplah untuk migrasi ke hosting berbasis cloud. Jika Anda mempertimbangkan atau berencana untuk bermigrasi ke AWS Cloud (alih-alih memutakhirkan perangkat keras Anda sendiri), solusi hosting hybrid adalah cara yang layak bagi Anda untuk melakukan transisi secara bertahap seperti yang Anda butuhkan.

- Tingkatkan latensi untuk pemain di lokasi di luar yang dilayani oleh Amazon GameLift Servers. Jika Anda sudah menggunakan hosting Amazon GameLift Servers terkelola, Anda mungkin perlu mendukung pemain dalam situasi tertentu. Misalnya, Anda mungkin ingin menjangkau pemain di lokasi terpencil yang tidak biasa atau secara signifikan mengurangi latensi ke area tersebut. Tambahkan lokasi hosting khusus dan gunakan Amazon GameLift Servers Anywhere untuk mengelola lokasi tersebut bersama dengan sumber daya hosting terkelola Anda.

Mulailah mengembangkan solusi hosting Amazon GameLift Servers hybrid untuk game Anda.

Fitur utama

- Gunakan klien game dan komponen server yang sama dengan sumber daya hosting yang dikelola dan dikelola sendiri. Berikan pengalaman pemain terpadu di semua sumber daya hosting.
- Gunakan FlexMatch mak comblang yang sama untuk menempatkan kecocokan di semua sumber daya hosting.
- Kelola sumber daya hosting hybrid Anda secara terpusat saat Anda menerapkannya di seluruh dunia.
- Saat permintaan pemain berfluktuasi, kelola beban sesi permainan dengan mulus di seluruh sumber daya yang dikelola dan dikelola sendiri.
- Dengan Amazon GameLift Servers Agen, Anda dapat menggunakan alat yang sama untuk mengelola siklus hidup server game pada semua jenis sumber daya hosting.
- Kumpulkan metrik dan log game dan pemain di semua sumber daya hosting. Manfaatkan Amazon GameLift Servers fitur dan AWS layanan lain untuk menggabungkan data dan mengembangkan solusi observabilitas yang kohesif.

Hosting di mana saja

Gunakan armada Amazon GameLift Servers Anywhere dengan manajemen sesi Amazon GameLift Servers game, termasuk perjudian, untuk meng-host server game kustom Anda di mana pun Anda mau. Anywherearmada sangat berguna sebagai lingkungan pengujian untuk pengembangan game yang cepat dan berulang. Siapkan Anywhere armada untuk stasiun kerja lokal Anda sendiri atau sekumpulan sumber daya hosting berbasis cloud. Untuk hosting produksi, Anda dapat menggunakan pendekatan hybrid dengan Anywhere armada untuk perangkat keras lokal Anda yang dilengkapi dengan Amazon GameLift Servers armada terkelola.

Untuk informasi selengkapnya tentang pengujian dengan Anywhere, lihat [Siapkan pengujian lokal dengan Amazon GameLift Servers Anywhere](#). Untuk informasi selengkapnya tentang menyiapkan armada Anywhere, lihat [Menyiapkan armada hosting dengan Amazon GameLift Servers](#).

[Mulailah mengembangkan solusi hosting Amazon GameLift Servers Anywhere untuk game Anda.](#)

Fitur utama

- Lakukan pengujian cepat dan berulang saat Anda mengembangkan game multipemain Anda.
- Gunakan Amazon GameLift Servers alat untuk mengelola server game yang di-host di perangkat keras Anda sendiri.
- Manfaatkan perangkat keras yang tersedia yang paling dekat dengan pemain Anda, di mana saja.

Amazon GameLift Servers FlexMatch untuk perijodohan

Gunakan FlexMatch untuk membangun set aturan khusus untuk menentukan pertandingan multipemain untuk game Anda. FlexMatch menggunakan set aturan untuk membandingkan pemain yang kompatibel untuk setiap pertandingan dan memberi pemain pengalaman multipemain yang ideal.

Untuk informasi lebih lanjut tentang FlexMatch, lihat [Apa itu Amazon GameLift Servers FlexMatch?](#)

Fitur utama

- Seimbangkan kecepatan pembuatan pertandingan dan kualitas kecocokan.
- Cocokkan pemain atau tim berdasarkan karakteristik yang ditentukan.
- Tentukan aturan untuk menempatkan pemain ke dalam pertandingan berdasarkan latensi.

Amazon GameLift Servers FleetIQ untuk hosting Amazon EC2 yang dikelola sendiri

Gunakan grup server FleetIQ game untuk bekerja secara langsung dengan sumber daya hosting Anda di Amazon EC2 dan Amazon EC2 Auto Scaling. Ini memberikan manfaat Amazon GameLift Servers pengoptimalan untuk hosting game yang murah dan tangguh. Solusi ini untuk pengembang game yang membutuhkan lebih banyak fleksibilitas daripada Amazon GameLift Servers solusi yang dikelola sepenuhnya.

Untuk informasi tentang cara FleetIQ bekerja dengan Amazon EC2 dan EC2 Auto Scaling untuk hosting game, lihat Panduan [Amazon GameLift ServersFleetIQPengembang](#).

Fitur utama

- Dapatkan penyeimbangan Instans Spot yang dioptimalkan menggunakan FleetIQ algoritme.
- Gunakan fitur perutean pemain untuk mengelola sumber daya server game Anda secara efisien, dan berikan pengalaman pemain yang lebih baik untuk bergabung dengan game.
- Secara otomatis menskalakan kapasitas hosting berdasarkan penggunaan pemain.
- Langsung mengelola EC2 instans Amazon sendiri Akun AWS.
- Gunakan salah satu format executable server game yang didukung, termasuk Windows, Linux, container, dan Kubernetes.

Amazon GameLift ServersRealtime dengan logika server yang dapat disesuaikan

Gunakan Realtime server untuk meng-host game yang tidak memerlukan server game yang dibuat khusus. Solusi server ringan ini menyediakan server game yang dapat Anda konfigurasi agar sesuai dengan game Anda. Anda dapat Realtime meng-host server menggunakan solusi hosting Amazon GameLift Servers terkelola.

Untuk informasi lebih lanjut tentang hosting dengan Amazon GameLift ServersRealtime, lihat [Mengintegrasikan game dengan Amazon GameLift ServersRealtime](#).

Fitur utama

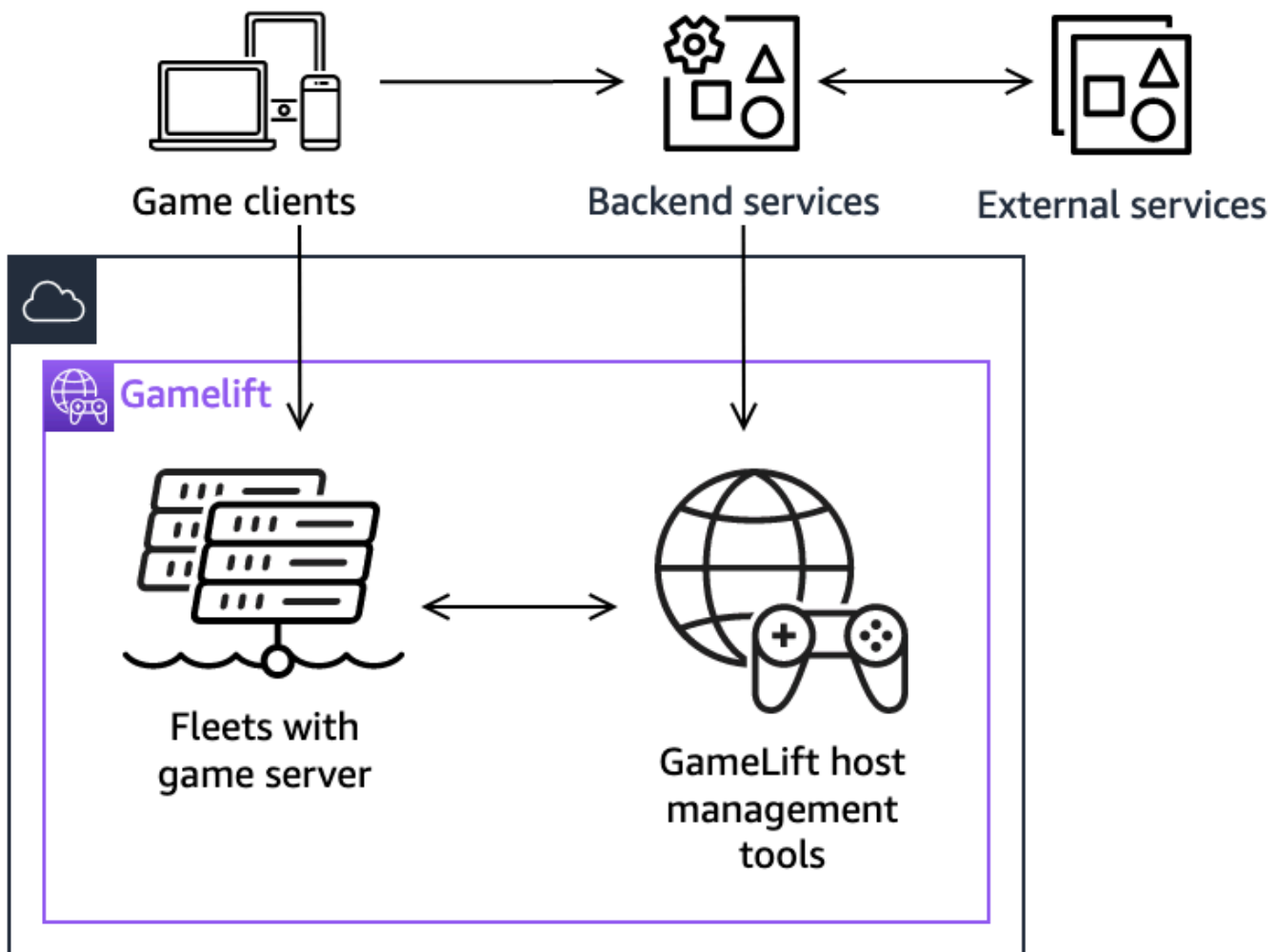
- Gunakan fitur Amazon GameLift Servers manajemen, termasuk penskalaan otomatis, antrian multi-lokasi, dan penempatan sesi game.
- Gunakan sumber daya Amazon GameLift Servers hosting dan pilih jenis perangkat keras AWS komputasi untuk armada Anda.
- Manfaatkan tumpukan jaringan lengkap untuk interaksi klien dan server game.
- Dapatkan fungsi server game inti dengan logika server yang dapat disesuaikan.
- Buat pembaruan langsung untuk Realtime konfigurasi dan logika server.

Arsitektur Amazon GameLift Servers solusi terkelola

Diagram dalam topik ini menguraikan bagaimana solusi hosting lengkap dengan Amazon GameLift Servers terstruktur.

Komponen game dengan hosting

Diagram berikut menggambarkan bagaimana komponen kunci dari solusi Amazon GameLift Servers hosting terkelola bekerja sama untuk menjalankan server game khusus dan membantu pemain menemukan dan terhubung ke sesi game yang dihosting. Solusi hosting yang Anda kembangkan untuk game Anda akan mencakup sebagian besar atau semua komponen ini.



Komponen kunci dari arsitektur ini meliputi:

Klien game

Klien game adalah perangkat lunak Anda yang berjalan di perangkat pemain. Pemain memainkan game Anda dengan bergabung dengan sesi permainan di server game yang dihosting. Klien game meminta untuk bergabung dengan sesi permainan melalui layanan backend, menerima informasi koneksi untuk sesi permainan, dan menggunakannya untuk terhubung langsung dengan sesi permainan. Untuk informasi selengkapnya, lihat [Mempersiapkan game untuk Amazon GameLift Servers](#). Saat menghubungkan ke Realtime server, klien game menggunakan SDK klien untuk Amazon GameLift ServersRealtime.

Layanan backend

Layanan backend adalah layanan khusus yang Anda buat untuk menangani komunikasi dengan Amazon GameLift Servers layanan atas nama klien game. Anda juga dapat menggunakan layanan backend untuk tugas-tugas khusus game seperti otentikasi dan otorisasi pemain, inventaris, atau kontrol mata uang. Layanan backend berkomunikasi dengan Amazon GameLift Servers layanan menggunakan operasi API di SDK. AWS

Layanan backend membuat permintaan untuk mendapatkan informasi sesi permainan yang ada dan untuk memulai sesi permainan. Permintaan untuk sesi permainan baru menentukan karakteristik tertentu, seperti jumlah maksimum pemain. Permintaan ini meminta Amazon GameLift Servers untuk memulai proses penempatan sesi permainan. Ketika sesi permainan siap menerima pemain, layanan backend mengambil informasi koneksi dan memberikannya kepada klien game.

Layanan eksternal

Game Anda dapat mengandalkan layanan eksternal, seperti untuk memvalidasi keanggotaan berlangganan. Layanan eksternal dapat meneruskan informasi ke server game Anda melalui layanan backend dan. Amazon GameLift Servers

Server game

Server game adalah perangkat lunak server game Anda yang berjalan pada serangkaian sumber daya hosting. Anda mengunggah perangkat lunak server game AndaAmazon GameLift Servers, yang menyebarkannya ke sumber daya hosting dan mulai menjalankan proses server. Setiap proses server game terhubung dengan Amazon GameLift Servers layanan untuk memberi sinyal kesiapan untuk menyelenggarakan sesi game. Ini berinteraksi dengan layanan untuk memulai sesi permainan, memvalidasi pemain yang baru terhubung, dan melaporkan status sesi permainan dan koneksi pemain.

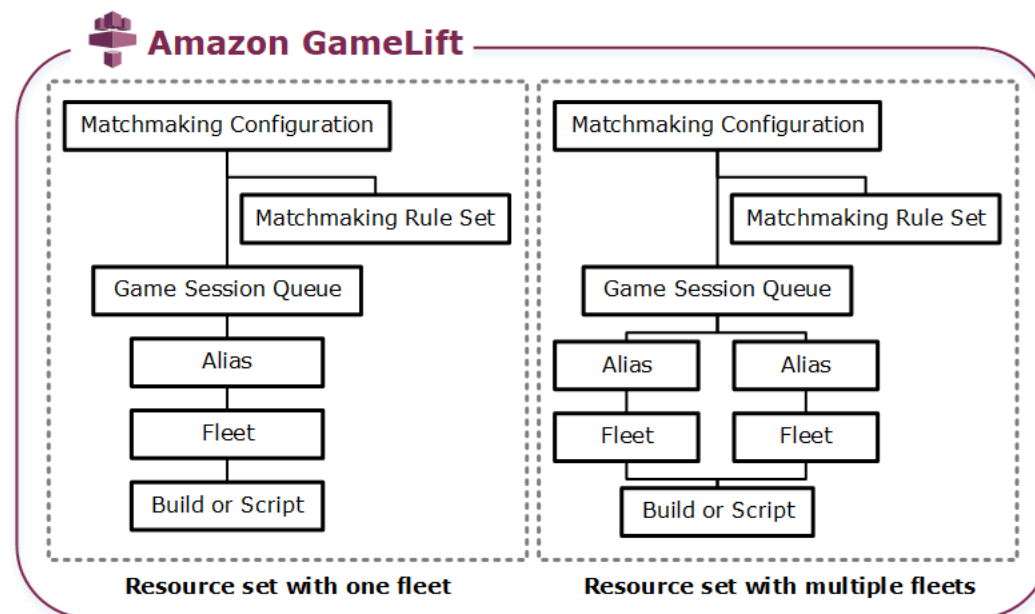
Server game khusus berkomunikasi Amazon GameLift Servers dengan menggunakan SDK server untuk Amazon GameLift Servers. Untuk informasi lebih lanjut, lihat [Integrasikan game dengan server game khusus](#). Realtime server adalah server game yang disediakan oleh Amazon GameLift Servers. Anda dapat menyesuaikan logika server dengan menyediakan skrip khusus. Untuk informasi selengkapnya, lihat [Mengintegrasikan game dengan Amazon GameLift Servers Realtime](#).

Alat manajemen host

Saat menyiapkan dan mengelola sumber daya hosting, pemilik game menggunakan alat manajemen hosting untuk mengelola pembuatan server game atau skrip, armada, perjuduhan, dan antrian. Amazon GameLift Servers Alat yang diatur dalam AWS SDK dan konsol menyediakan beberapa cara bagi Anda untuk mengelola sumber daya hosting Anda. Anda dapat mengakses server game individual dari jarak jauh untuk pemecahan masalah.

Sumber daya solusi hosting

Diagram berikut menggambarkan Amazon GameLift Servers sumber daya yang membentuk solusi hosting terkelola. Menyediakan pembuatan server khusus atau Amazon GameLift Servers Realtime skrip, menyebarkan armada komputasi untuk meng-host server game, dan kemudian mengatur antrian sesi permainan untuk menemukan sumber daya hosting yang tersedia dan memulai sesi permainan baru. Untuk game yang menggunakan FlexMatch perjuduhan, tambahkan konfigurasi perjuduhan dan aturan perjuduhan yang ditetapkan untuk menghasilkan pertandingan pemain.



Kode server game

- **Build** — Perangkat lunak server game yang dibuat khusus yang berjalan Amazon GameLift Servers dan menyelenggarakan sesi permainan untuk pemain Anda. Game build mewakili kumpulan file yang menjalankan server game Anda pada sistem operasi tertentu, dan yang harus Anda integrasikan Amazon GameLift Servers. Unggah file build game ke Amazon GameLift Servers Wilayah AWS tempat Anda berencana menyiapkan armada. Untuk informasi selengkapnya, lihat [Menerapkan build server khusus untuk Amazon GameLift Servers mengadakan](#).
- **Script** — Konfigurasi dan logika permainan kustom Anda untuk digunakan dengan Amazon GameLift Servers Realtime. Konfigurasi Amazon GameLift Servers Realtime untuk klien game Anda dengan membuat skrip menggunakan JavaScript, dan tambahkan logika game khusus untuk menyelenggarakan sesi game untuk pemain Anda. Untuk informasi selengkapnya, lihat [Menyebarkan skrip untuk Amazon GameLift Servers Realtime](#).

Armada

Kumpulan sumber daya komputasi yang menjalankan server game Anda dan menyelenggarakan sesi permainan untuk pemain Anda. Untuk informasi tentang tempat Anda dapat menyebarkan armada, lihat [Amazon GameLift Servers lokasi layanan](#). Untuk informasi tentang membuat armada, lihat [Menyiapkan armada hosting dengan Amazon GameLift Servers](#).

Alias

Pengidentifikasi abstrak untuk armada yang dapat Anda gunakan untuk mengubah armada yang terhubung dengan pemain Anda kapan saja. Untuk informasi selengkapnya, lihat [Buat sebuah Amazon GameLift Servers alias](#).

Antrian sesi permainan

Mekanisme penempatan sesi permainan yang menerima permintaan untuk sesi permainan baru dan mencari server game yang tersedia untuk menjadi tuan rumah sesi baru. Untuk informasi lebih lanjut tentang antrian sesi permainan, lihat [Mengelola penempatan sesi permainan dengan Amazon GameLift Servers mengantre](#).

Cara kerja Amazon GameLift Servers

Topik ini menjelaskan bagaimana Amazon GameLift Servers mengelola hosting khusus untuk server game multipemain Anda dan membuatnya tersedia untuk pemain. Ini menguraikan bagaimana fitur inti bekerja.

Server hosting game

Dengan Amazon GameLift Servers, Anda dapat meng-host server game Anda dengan beberapa cara berbeda: Dikelola Amazon GameLift Servers, Amazon GameLift Servers FleetIQ, dan Di Mana Amazon GameLift Servers Saja. Untuk informasi lebih lanjut tentang Amazon GameLift Servers FleetIQ, lihat [Apa itu Amazon GameLift Servers FleetIQ?](#)

Anda dapat merancang armada yang sesuai dengan kebutuhan game Anda. Untuk informasi lebih lanjut tentang merancang armada, lihat [Sesuaikan Amazon GameLift Servers EC2 armada yang dikelola](#).

Dikelola Amazon GameLift Servers

Dengan dikelola Amazon GameLift Servers, Anda dapat meng-host server game Anda pada sumber daya komputasi Amazon GameLift Servers virtual, yang disebut instance. Siapkan sumber daya hosting Anda dengan membuat armada instance dan menerapkannya untuk menjalankan server game Anda.

Amazon GameLift Servers Dimanapun

Dengan Amazon GameLift Servers Anywhere, Anda dapat meng-host server game Anda pada komputasi yang Anda kelola. Siapkan sumber daya hosting Anda dengan membuat armada Anywhere yang mereferensikan komputasi Anda.

Alias armada

Alias adalah sebutan yang dapat Anda transfer antar armada, menjadikannya cara yang nyaman untuk memiliki lokasi armada generik. Anda dapat menggunakan alias untuk mengalihkan klien game dari menggunakan satu armada ke armada lainnya tanpa mengubah klien game Anda. Anda juga dapat membuat alias terminal yang Anda arahkan ke konten.

Menjalankan sesi game

Setelah Anda menerapkan build server game ke armada dan Amazon GameLift Servers meluncurkan proses server game di setiap instance, armada dapat menghosting sesi game. Amazon GameLift Servers memulai sesi permainan baru ketika layanan klien game Anda mengirim permintaan penempatan ke layanan backend atau ke Amazon GameLift Servers

Penempatan sesi permainan dan FleetIQ algoritme

Antrian menggunakan FleetIQ algoritme untuk memilih server game yang tersedia untuk menjadi tuan rumah sesi permainan baru. Komponen kunci untuk penempatan sesi permainan adalah antrian sesi Amazon GameLift Servers permainan. Anda menetapkan antrian sesi permainan daftar armada, yang menentukan di mana antrian dapat menempatkan sesi permainan. Untuk informasi lebih lanjut tentang antrian sesi permainan dan cara mendesainnya untuk game Anda, lihat. [Kustomisasi antrian sesi permainan](#)

Mengoptimalkan penempatan sesi game dengan ping beacon UDP

Dengan menggunakan beacon ping Amazon GameLift Servers UDP, Anda dapat menghitung latensi pulang-pergi untuk paket UDP antara pemain dan server game di lokasi yang berbeda untuk membantu memilih lokasi optimal untuk sesi permainan. Untuk informasi selengkapnya tentang suar ping UDP dan bagaimana Anda dapat menggunakannya untuk mengukur latensi, lihat. [Suar ping UDP](#)

- Untuk game yang menggunakan antrian sesi game untuk penempatan, permintaan penempatan dapat menyertakan data latensi, yang secara otomatis digunakan antrian untuk memprioritaskan lokasi penempatan. Anda dapat menyesuaikan prioritas lebih lanjut dan dapat menyiapkan kebijakan termasuk batas nilai latensi. Lihat [Prioritaskan penempatan sesi permainan](#).
- Untuk game yang tidak menggunakan antrian sesi game tetapi memiliki armada dengan beberapa lokasi, Anda dapat mengevaluasi data latensi dan memilih lokasi terbaik yang tersedia sebelum membuat permintaan sesi game. Amazon GameLift Servers Lihat bagian Dapatkan dan Buat sesi permainan di [Tambahkan Amazon GameLift Servers ke klien game Anda](#).
- Jika Anda menggunakan FlexMatch untuk perjodohan, Anda dapat menetapkan aturan kecocokan untuk menggunakan data latensi. Lihat [Meminta perjodohan untuk pemain](#) dan bagian [Aturan latensi](#) dari jenis FlexMatch aturan di Panduan Pengembang. Amazon GameLift Servers FlexMatch

Koneksi pemain ke game

Sebagai bagian dari proses penempatan sesi permainan, antrian atau sesi permainan meminta server game yang dipilih untuk memulai sesi permainan baru. Server game merespons prompt dan melaporkan kembali Amazon GameLift Servers kapan siap menerima koneksi pemain. Amazon GameLift Servers kemudian mengirimkan informasi koneksi ke layanan backend atau layanan klien game. Klien game Anda menggunakan informasi ini untuk connect langsung ke sesi game dan memulai gameplay.

Penskalaan kapasitas armada

Saat armada aktif dan siap menjadi tuan rumah sesi permainan, Anda dapat menyesuaikan kapasitas armada Anda untuk memenuhi permintaan pemain. Kami menyarankan Anda menemukan keseimbangan antara semua pemain yang masuk menemukan permainan dengan cepat dan pengeluaran berlebihan pada sumber daya yang menganggur.

Amazon GameLift Servers menyediakan alat penskalaan otomatis yang sangat efektif, atau Anda dapat mengatur kapasitas armada secara manual. Untuk informasi selengkapnya, lihat [Menskalakan kapasitas hosting game dengan Amazon GameLift Servers](#).

Penskalaan otomatis

Amazon GameLift Servers menyediakan dua metode penskalaan otomatis:

- [Penskalaan otomatis berbasis target](#)
- [Skala otomatis dengan kebijakan berbasis aturan](#)

Fitur penskalaan tambahan

- Perlindungan sesi permainan — Mencegah Amazon GameLift Servers dari mengakhiri sesi permainan yang menjadi tuan rumah pemain aktif selama acara scale-down.
- Batas penskalaan – Mengontrol penggunaan instans secara keseluruhan dengan pengaturan batas minimum dan maksimum pada jumlah instans dalam armada.
- Menangguhkan penskalaan otomatis — Tangguhkan penskalaan otomatis di tingkat lokasi armada tanpa mengubah atau menghapus kebijakan penskalaan otomatis Anda.
- Metrik penskalaan — Lacak riwayat kapasitas dan peristiwa penskalaan armada.

Memantau Amazon GameLift Servers

Ketika Anda memiliki armada dan berjalan, Amazon GameLift Servers kumpulkan berbagai informasi untuk membantu Anda memantau kinerja server game yang Anda gunakan. Anda dapat menggunakan informasi ini untuk mengoptimalkan penggunaan sumber daya, memecahkan masalah, dan mendapatkan wawasan tentang bagaimana pemain aktif dalam game Anda. Amazon GameLift Servers mengumpulkan yang berikut ini:

- Armada, lokasi, sesi permainan, dan detail sesi pemain

- Metrik penggunaan
- Kesehatan proses server
- Log sesi permainan

Untuk informasi lebih lanjut tentang pemantauan di Amazon GameLift Servers, lihat [Pemantauan Amazon GameLift Servers](#).

Menggunakan AWS sumber daya lain

Server dan aplikasi game Anda dapat berkomunikasi dengan AWS sumber daya lain. Misalnya, Anda mungkin menggunakan seperangkat layanan web untuk autentikasi pemain atau jaringan. Agar server game Anda dapat mengakses AWS sumber daya yang Anda kelola, izinkan secara eksplisit Amazon GameLift Servers untuk mengakses sumber daya AWS.

Amazon GameLift Servers menyediakan beberapa opsi untuk mengelola jenis akses ini. Lihat informasi yang lebih lengkap di [Berkomunikasi dengan AWS sumber daya lain dari armada Anda](#).

Cara kerja kontainer Amazon GameLift Servers

Amazon GameLift Servers Armada kontainer dirancang untuk memberi Anda fleksibilitas dalam cara Anda menerapkan dan menskalakan aplikasi kontainer Anda. Ini menggunakan Amazon Elastic Container Service (Amazon ECS) untuk mengelola penerapan tugas dan eksekusi untuk armada Anda. Amazon GameLift Servers Topik ini menjelaskan elemen struktural dasar untuk menjalankan kontainer pada armada yang Amazon GameLift Servers dikelola, menggambarkan arsitektur umum, dan menguraikan beberapa konsep inti.

 Percepat orientasi dengan alat ini untuk kontainer terkelola:

- [Kit starter kontainer](#) merampingkan integrasi dan pengaturan armada. Ini menambahkan fitur manajemen sesi permainan penting ke server game Anda, dan menggunakan templat yang telah dikonfigurasi sebelumnya untuk membangun armada kontainer dan pipeline penyebaran otomatis untuk server game Anda. Setelah penerapan, gunakan Amazon GameLift Servers konsol dan alat API untuk memantau kinerja armada, mengelola sesi game, dan menganalisis metrik.
- Untuk pengembang Unreal Engine atau Unity, gunakan [Amazon GameLift Servers plugin](#) untuk mengintegrasikan server game Anda dan membangun armada kontainer dari dalam lingkungan pengembangan mesin game Anda. Alur kerja terpandu plugin membantu Anda

membuat solusi yang cepat dan sederhana dengan hosting berbasis cloud menggunakan kontainer terkelola. Kemudian bangun di atas fondasi ini untuk membuat solusi hosting khusus untuk game Anda.

Komponen armada kontainer

Armada

Armada kontainer adalah kumpulan EC2 instance Amazon untuk meng-host server game kontainer Anda. Instans ini dikelola oleh Amazon GameLift Servers atas nama Anda. Saat membuat armada, Anda mengonfigurasi bagaimana arsitektur kontainer dengan perangkat lunak server game Anda diterapkan ke setiap instance armada. Anda dapat membuat armada kontainer dengan instance di satu atau beberapa lokasi geografis. Anda dapat menggunakan alat Amazon GameLift Servers penskalaan untuk secara otomatis menskalakan kapasitas armada kontainer untuk menyelenggarakan sesi permainan dan pemain.

Instans

EC2 Instans Amazon adalah server virtual yang menyediakan kapasitas komputasi untuk hosting game. Dengan Amazon GameLift Servers, Anda dapat memilih dari berbagai jenis instans. Setiap jenis instans menawarkan kombinasi CPU, memori, penyimpanan, dan kapasitas jaringan yang berbeda.

Saat Anda membuat armada kontainer, Amazon GameLift Servers gunakan kontainer Anda berdasarkan jenis instans yang Anda pilih dan konfigurasi armada Anda. Setiap instance armada yang digunakan identik dan menjalankan perangkat lunak server game kontainer Anda dengan cara yang sama. Jumlah instance dalam armada menentukan ukuran armada dan kapasitas hosting game.

Kelompok kontainer

Amazon GameLift Servers menggunakan konsep grup kontainer untuk mendeskripsikan dan mengelola satu set kontainer. Sebuah grup kontainer mirip dengan container “task” atau “pod”. Dalam setiap grup kontainer, Anda dapat menentukan bagaimana kontainer berperilaku, mengatur dependensi, dan berbagi sumber daya CPU dan memori yang tersedia.

Setiap instance armada dapat memiliki jenis grup kontainer berikut:

- Grup kontainer server game mengelola kontainer yang menjalankan aplikasi server game dan perangkat lunak pendukung Anda. Armada kontainer harus memiliki salah satu dari jenis grup

kontainer ini untuk menjadi tuan rumah sesi permainan dan pemain. Grup kontainer server game dapat direplikasi di seluruh instance armada. Jumlah replika grup server game per instance armada tergantung pada persyaratan komputasi perangkat lunak Anda dan sumber daya komputasi yang tersedia pada instance tersebut.

- Grup kontainer per-instance, yang bersifat opsional, memberi Anda kemampuan untuk menjalankan perangkat lunak tambahan pada setiap instance armada. Mereka berguna untuk menjalankan layanan latar belakang atau program utilitas, seperti untuk pemantauan. Perangkat lunak server game Anda tidak secara langsung bergantung pada proses dalam grup per-instance. Hanya satu salinan grup kontainer per-instance yang disebarkan ke setiap instance armada.

Setiap kelompok kontainer dalam armada kontainer memiliki satu kontainer yang ditunjuk “penting”. Kontainer penting mendorong siklus hidup grup kontainer. Jika wadah penting gagal, seluruh grup kontainer akan dimulai ulang.

Kontainer

Wadah adalah elemen paling dasar dari arsitektur berbasis kontainer. Ini termasuk gambar kontainer dengan perangkat lunak yang dapat dieksekusi dan file dependen. Tentukan wadah untuk mengonfigurasi bagaimana perangkat lunak berjalan dan berinteraksi dengannya Amazon GameLift Servers.

Amazon GameLift Servers mendefinisikan dua jenis kontainer:

- Wadah server game mencakup semua yang Anda butuhkan untuk menjalankan proses server game Anda dan menyelenggarakan sesi permainan untuk pemain. Ini termasuk pembuatan server game Anda dan perangkat lunak dependen. Tentukan satu wadah server game untuk grup kontainer server game armada. Wadah server game secara otomatis dianggap penting untuk grup kontainer.
- Kontainer dukungan menjalankan perangkat lunak tambahan untuk mendukung server game Anda. Ini mirip dengan konsep wadah “sespan”. Ini memberi Anda opsi untuk menjalankan dan menskalakan perangkat lunak pendukung di samping server game Anda tetapi mengelola sebagai wadah terpisah. Dalam grup kontainer server game, Anda dapat menentukan nol atau lebih kontainer dukungan. Dalam grup kontainer per-instance, semua kontainer adalah kontainer pendukung. Setiap wadah pendukung dapat ditunjuk penting.

Komputasi

Komputasi mewakili salinan grup kontainer server game pada instance armada.

Arsitektur umum

Diagram berikut menggambarkan struktur armada kontainer yang paling sederhana. Dalam struktur ini, setiap instance dalam armada memelihara satu salinan grup kontainer server game. Grup kontainer memiliki wadah server game tunggal yang menjalankan satu proses server game. Dalam contoh ini, armada kontainer dikonfigurasi untuk menempatkan satu salinan grup kontainer server game per instance. Dengan arsitektur ini, setiap instance menjalankan satu proses server game.

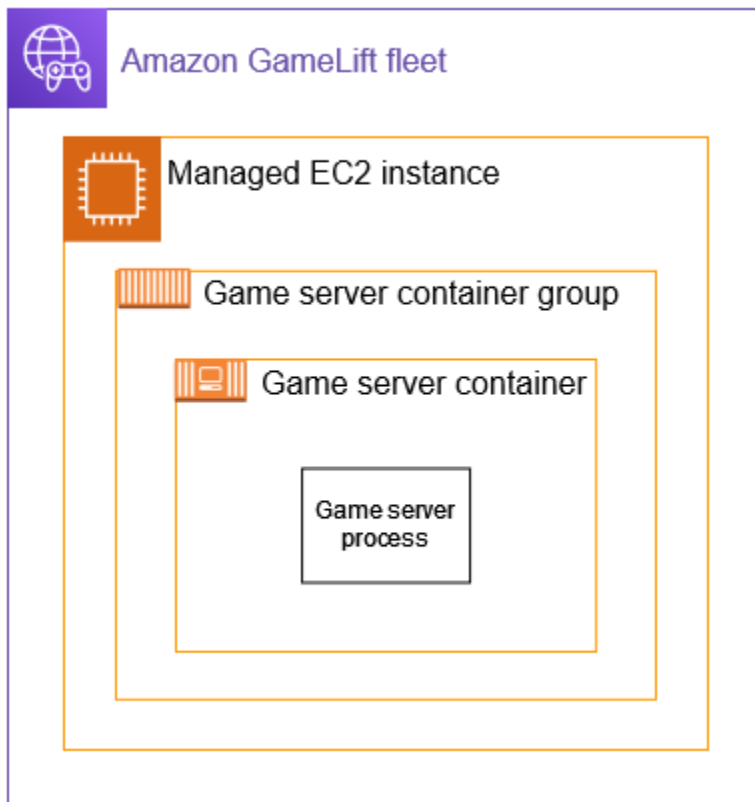
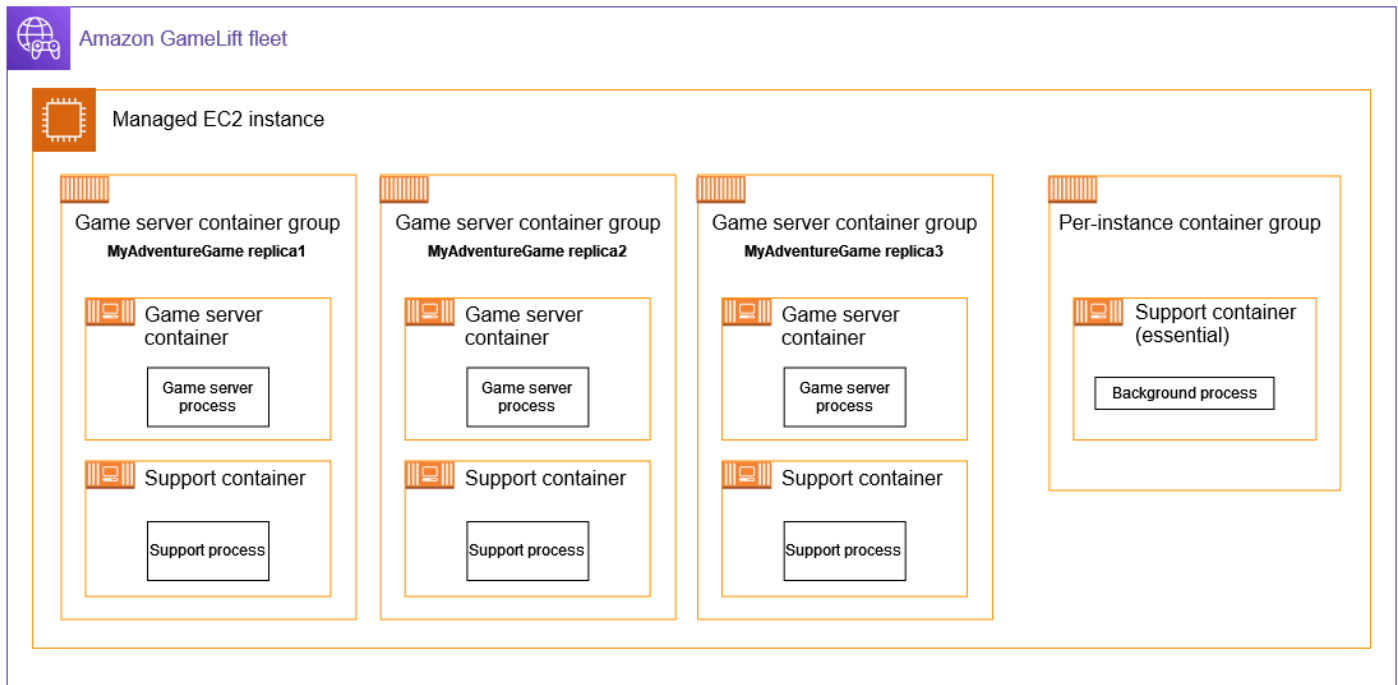


Diagram kedua ini menggambarkan arsitektur armada kontainer yang lebih kompleks. Dalam struktur ini, armada memiliki grup kontainer server game dan grup kontainer per-instance. Grup kontainer server game memiliki wadah terpisah untuk proses server game dan proses dukungan. Armada dikonfigurasi untuk menempatkan tiga salinan grup kontainer server game pada setiap instance armada. Grup kontainer per-instance tidak pernah direplikasi. Dalam contoh ini, armada kontainer dikonfigurasi untuk menempatkan tiga salinan grup kontainer server game per instance. Dengan arsitektur ini, setiap instance menjalankan tiga proses server game.



Fitur inti

Bagian ini merangkum bagaimana Amazon GameLift Servers mengimplementasikan beberapa konsep kontainer dasar. Untuk petunjuk tentang cara bekerja dengan armada kontainer, lihat topik yang relevan dalam panduan ini.

Pembaruan armada aktif

Kontainer terkelola menyediakan dukungan lanjutan untuk membantu Anda mengelola siklus hidup perangkat lunak dan arsitektur kontainer yang dihosting. Anda dapat memperbarui definisi kontainer, termasuk gambar kontainer, dan menerapkan perubahan ke armada yang ada. Fitur ini membuatnya lebih cepat dan lebih mudah untuk mengulangi perubahan pada kontainer Anda selama pengembangan. Ini juga menyediakan fitur untuk membantu Anda membangun, menyebarkan, dan melacak pembaruan versi perangkat lunak Anda dari waktu ke waktu. Fitur ini mencakup:

- Kelola pembaruan dan pembuatan versi definisi grup kontainer. Anda dapat memperbarui hampir semua properti definisi grup kontainer, termasuk gambar kontainer dan pengaturan konfigurasi. Setiap kali Anda memperbarui wadah, Amazon GameLift Servers secara otomatis menetapkan nomor versi ke pembaruan dan secara default mempertahankan semua versi. Anda dapat mengakses versi tertentu, dan Anda dapat menghapus versi yang diinginkan. Saat membuat armada kontainer, Anda dapat menentukan definisi dan versi grup kontainer yang akan diterapkan.

- Perbarui armada kontainer yang ada dengan definisi grup kontainer baru dan pengaturan konfigurasi. Anda dapat menerapkan pembaruan kontainer ke armada yang sudah disebar ke instance armada. Anda dapat melacak status penerapan pembaruan di setiap lokasi armada menggunakan AWS Management Console atau AWS SDK dan CLI.
- Konfigurasikan bagaimana Anda ingin pembaruan armada diterapkan di seluruh armada aktif.
 - Perlindungan sesi permainan. Pilih untuk melindungi instance armada dengan sesi permainan aktif hingga setelah sesi permainan berakhir (penerapan aman). Atau pilih untuk mengganti instance armada terlepas dari aktivitas sesi game (penerapan tidak aman). Gunakan penerapan yang tidak aman selama fase pengembangan dan pengujian untuk mengurangi waktu penerapan.
 - Persentase sehat minimum. Tentukan persentase tugas sehat yang ingin Anda pertahankan selama penerapan. Fitur ini memungkinkan Anda memutuskan berapa banyak instance armada yang terpengaruh selama penerapan. Nilai rendah memprioritaskan kecepatan penerapan, sementara nilai tinggi memastikan bahwa ketersediaan server game tetap tinggi selama penerapan.
 - Strategi kegagalan penerapan. Tentukan tindakan apa yang harus diambil jika penerapan gagal. Kegagalan penerapan berarti bahwa beberapa kontainer yang diperbarui telah gagal memeriksa status dan dianggap terganggu. Anda dapat mengatur penerapan untuk secara otomatis memutar kembali semua instance armada ke status yang digunakan sebelumnya. Atau Anda dapat memilih untuk mempertahankan beberapa instance armada yang terganggu untuk digunakan dalam debugging.

Kemampuan untuk memperbarui armada aktif sangat berguna ketika Anda ingin menyebarkan pembaruan ke perangkat lunak server game Anda. Setelah Anda membuat image kontainer baru untuk server game Anda, menerapkannya adalah proses dua langkah: pertama, perbarui definisi grup kontainer dengan gambar baru, dan kedua, perbarui armada kontainer. Amazon GameLift Servers menangani semua tugas lain sesuai kebutuhan.

Kemasan kontainer

Saat mengembangkan struktur kontainer Anda untuk penyebaran dalam armada kontainer, tujuan umum adalah mengoptimalkan penggunaan daya komputasi yang tersedia. Untuk mencapai tujuan ini, Anda ingin mengemas grup kontainer server game sebanyak mungkin ke setiap instance armada.

Amazon GameLift Servers membantu Anda melakukan ini dengan menghitung grup kontainer server game maksimum per instance, berdasarkan informasi berikut:

- Jenis instans armada dan vCPU dan sumber daya memorinya.
- Persyaratan vCPU dan memori untuk semua kontainer dalam grup kontainer server game.

Persyaratan vCPU dan memori untuk semua kontainer dalam grup kontainer per-instance, jika ada.

Saat Anda membuat armada kontainer, Anda dapat menggunakan maksimum yang dihitung atau Anda dapat menentukan nomor yang diinginkan. Sebagai praktik terbaik, rencanakan untuk bereksperimen dengan perangkat lunak server game kontainer Anda untuk menentukan persyaratan sumber daya untuk kinerja server game yang optimal.

Penskalaan kapasitas

Kapasitas armada mengukur jumlah sesi permainan yang dapat diselenggarakan armada secara bersamaan. Anda juga dapat mengukur kapasitas berdasarkan jumlah pemain bersamaan yang dapat didukung armada. Untuk menambah atau mengurangi kapasitas hosting armada, Anda menambah atau menghapus instance armada.

Armada kontainer dikonfigurasi untuk menjalankan sejumlah proses server game bersamaan pada setiap instance armada. (Anda dapat menghitung ini berdasarkan (1) grup kontainer server game per instance dan (2) jumlah proses server game yang berjalan di setiap grup kontainer.) Jumlah server game cncurrent per instance memberi tahu Anda apa dampaknya menambahkan atau menghapus setiap instance armada. Misalnya, jika armada kontainer Anda menjalankan 1 proses server game di setiap grup kontainer server game, dan setiap instance armada menampung 100 grup kontainer server game, Anda menambah atau mengurangi kapasitas armada Anda untuk meng-host sesi game bersamaan dengan penambahan 100. Jika setiap sesi permainan memiliki 10 slot pemain, maka Anda menambah atau mengurangi kapasitas armada Anda untuk menampung pemain dengan penambahan 1000.

Dengan armada kontainer, Anda dapat menggunakan salah satu metode penskalaan kapasitas yang disediakan oleh Amazon GameLift Servers. Ini termasuk:

- Atur kapasitas armada secara manual dengan menetapkan jumlah instans armada yang diinginkan.
- Siapkan penskalaan otomatis dengan menargetkan buffer yang diinginkan dari instance yang tersedia (pelacakan target). Metode ini secara otomatis mempertahankan sejumlah sumber daya hosting idle sehingga pemain yang masuk dapat masuk ke game dengan cepat. Saat permintaan pemain meningkat atau menurun, ukuran buffer ini terus disesuaikan.

- Siapkan penskalaan otomatis dengan aturan penskalaan khusus (fitur lanjutan). Metode ini memungkinkan Anda menskalakan berdasarkan metrik armada yang Anda pilih.

client/server Koneksi game

Dengan armada Amazon GameLift Servers terkelola, klien game terhubung langsung ke server game yang dihosting cloud Anda. Ketika klien game meminta untuk bergabung dengan game, Amazon GameLift Servers temukan sesi permainan dan berikan informasi koneksi (IP dan port) ke klien game. Anda dapat mengontrol akses eksternal ke instance armada dengan membuka rentang port tertentu (izin masuk) untuk armada. Izin masuk menentukan port mana yang terbuka untuk lalu lintas masuk. Anda dapat dengan cepat mematikan semua port, membatasi beberapa, atau membuka semua port.

Armada kontainer terkelola memerlukan pengaturan tambahan yang memungkinkan akses ke proses yang berjalan dalam wadah. Saat Anda membuat definisi kontainer, Anda menentukan satu set port, satu untuk setiap proses yang mengambil koneksi. Hal ini mencakup:

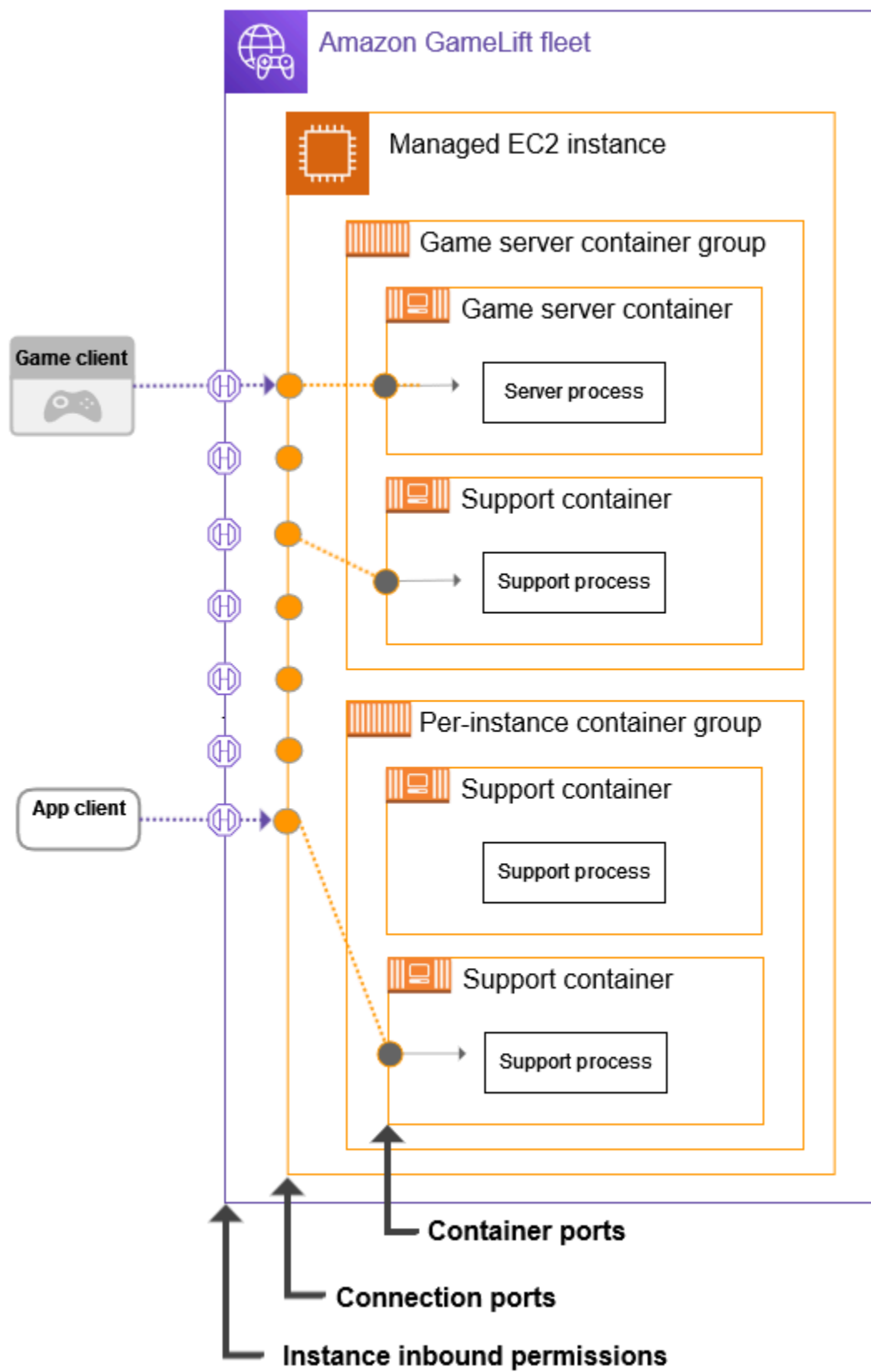
- Semua proses server game yang akan berjalan secara bersamaan di wadah server game. Semua proses server game harus memungkinkan klien game terhubung untuk bergabung dengan sesi game.
- Setiap proses dalam wadah pendukung yang perlu dihubungkan oleh sumber eksternal. Misalnya, Anda dapat terhubung dari jarak jauh ke aplikasi pengujian.

Saat Anda mengatur pengaturan port kontainer yang menghadap ke internal, Amazon GameLift Servers gunakan untuk menghitung izin masuk menghadap eksternal yang dapat dihubungkan oleh klien game dan aplikasi lain. Amazon GameLift Servers juga mengelola pemetaan antara izin masuk dan port kontainer individu yang memberi pemain akses ke sesi permainan dalam wadah. Pemetaan internal ini memberikan lapisan keamanan dengan melindungi server game Anda dari akses langsung ke port kontainer. Anda memiliki opsi untuk menyesuaikan pengaturan port yang menghadap ke luar armada sesuai kebutuhan. Untuk informasi selengkapnya tentang pengaturan port armada kontainer secara manual, lihat [Konfigurasi koneksi jaringan](#).

Anda dapat memodifikasi pengaturan port armada kontainer kapan saja. Perubahan ini memerlukan penerapan pembaruan armada.

Diagram berikut menggambarkan peran koneksi pelabuhan melintasi armada kontainer. Seperti yang ditunjukkan, Anda mengatur port pada kontainer individual, dan Amazon GameLift Servers menggunakan informasi ini untuk mengonfigurasi port yang cukup pada instance armada untuk memetakan ke setiap port kontainer. Baik izin masuk instans yang menghadap ke luar dan port

koneksi yang menghadap ke internal dihitung untuk armada Anda, kecuali jika Anda memilih Amazon GameLift Servers untuk mengaturnya secara manual.



Penebangan kontainer

Dalam armada kontainer terkelola, aliran keluaran standar (dan kesalahan standar) ditangkap untuk semua kontainer. Ini termasuk log sesi permainan server game Anda. Anda dapat mengonfigurasi armada kontainer untuk menggunakan salah satu dari beberapa opsi untuk menangani aliran keluaran:

- Simpan output kontainer sebagai aliran CloudWatch log Amazon. Setiap aliran log mereferensikan ID armada dan kontainer. Jika Anda memilih opsi pencatatan ini untuk armada, Anda menentukan grup CloudWatch log, yang mengatur semua aliran log dari armada. Anda kemudian dapat menggunakan CloudWatch fitur untuk mencari dan menganalisis data log sesuai kebutuhan.
- Simpan output kontainer ke bucket penyimpanan Amazon Simple Storage Service (Amazon S3). Anda dapat melihat, berbagi, atau mengunduh konten sesuai kebutuhan.
- Matikan logging. Dalam skenario ini, output kontainer tidak disimpan.

Amazon GameLift Servers mengirimkan data log dari armada kontainer terkelola ke CloudWatch atau layanan Amazon S3 di AWS akun Anda. Untuk melihat data Anda, gunakan AWS Management Console atau alat lainnya dengan masuk ke AWS akun Anda dan bekerja dengan layanan individual. Anda memperluas akses terbatas Amazon GameLift Servers untuk mengambil tindakan ini dengan membuat peran layanan untuk armada kontainer Anda.

Anda dapat memodifikasi konfigurasi pencatatan armada kontainer kapan saja. Perubahan ini memerlukan penerapan pembaruan armada.

Armada kontainer dan Agen Amazon GameLift Servers

Arsitektur kontainer yang umum digunakan menjalankan satu proses di setiap kontainer. Dalam armada Amazon GameLift Servers kontainer, grup kontainer server game memiliki satu wadah server game, yang menjalankan satu proses server game. Dengan arsitektur ini, Amazon GameLift Servers mengelola siklus hidup proses server game tunggal di setiap grup kontainer server game pada instance armada.

Jika Anda memilih untuk membangun arsitektur kontainer yang menjalankan beberapa proses server game di setiap grup kontainer server game, Anda memerlukan cara untuk mengelola siklus hidup semua proses. Ini termasuk tugas-tugas seperti memulai, mematikan, dan mengganti proses sesuai kebutuhan, mengelola sejumlah proses yang diinginkan untuk dijalankan secara bersamaan, dan menangani status kegagalan.

Anda dapat memilih untuk menggunakan Amazon GameLift Servers Agen untuk tugas-tugas ini. Untuk armada kontainer, Agen mengimplementasikan instruksi runtime yang menentukan executable mana yang akan dijalankan (dan berapa banyak), menyediakan parameter peluncuran, dan menetapkan aturan seputar aktivasi server game. Misalnya, instruksi runtime mungkin memberi tahu Agen untuk mempertahankan sepuluh proses server game untuk penggunaan produksi, dan satu proses server game dengan parameter peluncuran khusus untuk pengujian.

Untuk menggunakan Agen dengan armada kontainer Anda, tambahkan Agen ke gambar kontainer Anda dan sertakan satu set instruksi runtime. Untuk informasi lebih lanjut tentang Agen, lihat [Bekerja dengan Amazon GameLift Servers Agen](#).

Bagaimana pemain terhubung ke game

Sesi permainan adalah contoh dari permainan Anda yang sedang berjalan Amazon GameLift Servers. Untuk memainkan game Anda, pemain dapat menemukan dan bergabung dengan sesi permainan yang ada atau membuat sesi permainan baru dan bergabung dengannya. Seorang pemain bergabung dengan membuat sesi pemain untuk sesi permainan. Jika sesi permainan terbuka untuk pemain, maka Amazon GameLift Servers pesan slot untuk pemain dan berikan informasi koneksi. Pemain kemudian dapat connect ke sesi game dan mengklaim slot yang dicadangkan.

Untuk informasi terperinci tentang membuat dan mengelola sesi permainan dan sesi pemain dengan server game khusus, lihat [Tambahkan Amazon GameLift Servers ke klien game Anda](#). Untuk informasi tentang menghubungkan pemain Amazon GameLift Servers Realtime, lihat [Mengintegrasikan klien game untuk Amazon GameLift Servers Realtime](#).

Amazon GameLift Servers menyediakan beberapa fitur yang terkait dengan sesi permainan dan pemain.

Selenggarakan sesi permainan tentang sumber daya terbaik yang tersedia di beberapa lokasi

Pilih dari beberapa opsi saat mengonfigurasi cara Amazon GameLift Servers memilih sumber daya untuk menyelenggarakan sesi permainan baru. Jika Anda menjalankan armada di beberapa lokasi, maka Anda dapat merancang antrian sesi permainan yang menempatkan sesi permainan baru di armada apa pun terlepas dari lokasinya. Gunakan beacon ping UDP untuk menghitung data latensi, yang secara otomatis digunakan antrian untuk memprioritaskan lokasi penempatan.

Kontrol akses pemain ke sesi permainan

Konfigurasikan sesi permainan untuk mengizinkan atau menolak permintaan bergabung dari pemain baru, terlepas dari jumlah pemain yang terhubung.

Gunakan data game dan pemain khusus

Tambahkan data khusus ke objek sesi game dan objek sesi pemain. Amazon GameLift Servers meneruskan data sesi game ke server game saat memulai sesi permainan baru. Amazon GameLift Servers meneruskan data sesi pemain ke server game saat pemain terhubung ke sesi permainan.

Filter dan urutkan sesi permainan yang tersedia

Gunakan pencarian sesi dan urutkan untuk menemukan kecocokan terbaik untuk calon pemain, atau biarkan pemain memilih dari daftar sesi permainan yang tersedia. Gunakan pencarian sesi dan urutkan untuk menemukan sesi permainan berdasarkan karakteristik sesi. Anda juga dapat mencari dan mengurutkan berdasarkan data game kustom Anda sendiri.

Lacak data penggunaan game dan pemain

Secara otomatis memiliki log yang disimpan untuk sesi permainan yang diselesaikan. Anda dapat mengatur penyimpanan log saat mengintegrasikan Amazon GameLift Servers ke server game Anda. Untuk informasi selengkapnya, lihat [Mencatat pesan server di Amazon GameLift Servers](#).

Gunakan Amazon GameLift Servers konsol untuk melihat informasi terperinci tentang sesi permainan, termasuk metadata sesi, pengaturan, dan data sesi pemain. Lihat informasi yang lebih lengkap di [Sesi permainan dan pemain di Amazon GameLift Servers konsol](#) dan [Metrik](#).

Amazon GameLift Servers lokasi layanan

Amazon GameLift Servers fitur tersedia di beberapa Wilayah AWS dan Local Zones. Anda dapat merancang solusi hosting yang menempatkan server game Anda tepat di tempat pemain Anda berada.

AWS Lokasi yang didukung

Tabel berikut adalah daftar Wilayah AWS dan Local Zones yang mendukung Amazon GameLift Servers sumber daya. Ini menunjukkan jenis sumber daya yang dapat Anda buat di setiap lokasi.

Lokasi geografis	Kode lokasi	Beranda Wilayah untuk armada yang dikelola (lokasi tunggal)	Beranda Wilayah untuk armada yang dikelola (multi-lokasi)	Lokasi terpencil untuk armada terkelola (multi-lokasi)	Armada di mana saja	Antrean sesi game	FlexMatch mak comblang dan set aturan
US East (Northern Virginia)	us-east-1	Ya	Ya	Ya	Ya	Ya	Ya
US East (Ohio)	us-east-2	Ya		Ya	Ya	Ya	
US West (N. California)	us-west-1	Ya		Ya	Ya	Ya	
US West (Oregon)	us-west-2	Ya	Ya	Ya	Ya	Ya	Ya
Afrika (Cape Town)	af-south-1			Ya			
Asia Pasifik (Thailand)	ap-tenggara-7			Ya			
Asia Pasifik (Hong Kong)	ap-east-1			Ya			
Asia Pasifik (Malaysia)	ap-southeast-5			Ya			
Asia Pacific (Mumbai)	ap-south-1	Ya		Ya	Ya	Ya	

Lokasi geografis	Kode lokasi	Beranda Wilayah untuk armada yang dikelola (lokasi tunggal)	Beranda Wilayah untuk armada yang dikelola (multi-lokasi)	Lokasi terpencil untuk armada terkelola (multi-lokasi)	Armada di mana saja	Antrean sesi game	FlexMatch mak comblang dan set aturan
Asia Pacific (Osaka)	ap-northeast-3			Ya			
Asia Pacific (Seoul)	ap-northeast-2	Ya	Ya	Ya	Ya	Ya	Ya
Asia Pacific (Singapore)	ap-southeast-1	Ya		Ya	Ya	Ya	
Asia Pacific (Sydney)	ap-southeast-2	Ya	Ya	Ya	Ya	Ya	Ya
Asia Pacific (Tokyo)	ap-northeast-1	Ya	Ya	Ya	Ya	Ya	Ya
Canada (Central)	ca-central-1	Ya		Ya	Ya	Ya	
Europa (Frankfurt)	eu-central-1	Ya	Ya	Ya	Ya	Ya	Ya
Europa (Irlandia)	eu-west-1	Ya	Ya	Ya	Ya	Ya	Ya

Lokasi geografis	Kode lokasi	Beranda Wilayah untuk armada yang dikelola (lokasi tunggal)	Beranda Wilayah untuk armada yang dikelola (multi-lokasi)	Lokasi terpencil untuk armada terkelola (multi-lokasi)	Armada di mana saja	Antrean sesi game	FlexMatch mak comblang dan set aturan
Eropa (London)	eu-west-2	Ya		Ya	Ya	Ya	
Eropa (Milan)	eu-south-1			Ya			
Europe (Paris)	eu-west-3			Ya			
Eropa (Stockholm)	eu-north-1			Ya			
Timur Tengah (Bahrain)	me-south-1			Ya			
South America (São Paulo)	sa-east-1	Ya		Ya	Ya	Ya	
Zona lokal Atlanta	kami-timur-1-atl-1			Ya			
Zona lokal Chicago	us-timur-1-chi-1			Ya			
Zona lokal Dallas	us-timur-1-dfw-1			Ya			
Zona lokal Denver	us-barat-2-den-1			Ya			

Lokasi geografis	Kode lokasi	Beranda Wilayah untuk armada yang dikelola (lokasi tunggal)	Beranda Wilayah untuk armada yang dikelola (multi-lokasi)	Lokasi terpencil untuk armada terkelola (multi-lokasi)	Armada di mana saja	Antrean sesi game	FlexMatch mak comblang dan set aturan
Zona lokal Houston	kami-timur-1-iah-1			Ya			
Zona lokal Kota Kansas	us-timur-1-mci-1			Ya			
Zona lokal Los Angeles	us-west-2-lax-1			Ya			
Zona lokal Phoenix	kami-barat-2-phx-1			Ya			
Lagos, zona lokal Nigeria	af-selatan-1-los-1			Ya			

Note

Tidak semua Wilayah AWS diaktifkan secara default untuk file Akun AWS. Jika Anda menginginkan armada multi-lokasi dengan instance di Wilayah ini, Anda harus mengaktifkannya. Untuk informasi selengkapnya tentang Wilayah yang tidak diaktifkan secara default dan cara mengaktifkannya, lihat [Mengelola Wilayah AWS](#) di Referensi Umum AWS. Armada yang Anda buat sebelum 28 Februari 2022 tidak terpengaruh.

Selain itu, Anda harus memperbarui kebijakan Amazon GameLift Servers administrator Anda untuk mengizinkan `ec2:DescribeRegions` tindakan. Untuk contoh kebijakan dengan Wilayah yang tidak diaktifkan secara default, lihat [Contoh izin administrasi](#).

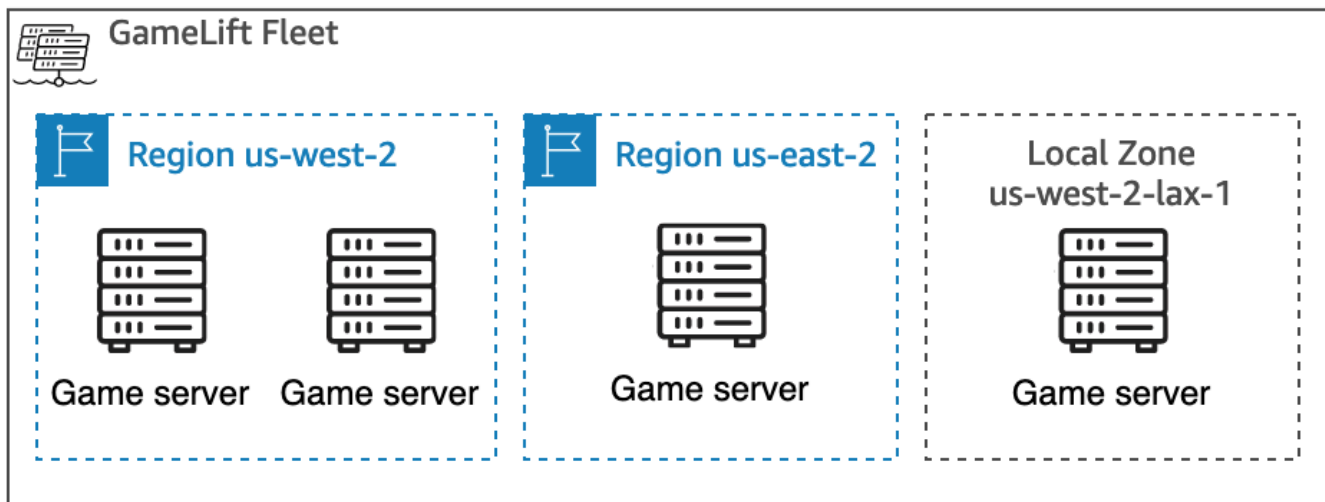
Lokasi untuk hosting terkelola

Amazon GameLift Servers hosting terkelola menyebarkan armada sumber daya server game. Setiap armada dibuat di Wilayah AWS, yang merupakan wilayah asal armada. Wilayah asal armada direferensikan dalam Nomor Sumber Daya Amazon (ARN) armada.

Anda dapat menyebarkan armada satu wilayah, dengan sumber daya hosting di wilayah asal saja. Atau, Anda dapat menyebarkan armada multi-lokasi, dengan sumber daya hosting di beberapa lokasi geografis. Armada multi-lokasi memiliki wilayah asal dan satu atau lebih lokasi terpencil. Saat mengelola kapasitas hosting untuk armada multi-lokasi, Anda dapat mengatur kapasitas untuk setiap lokasi secara individual.

Lokasi terpencil untuk armada multi-lokasi dapat berupa Local Zones lain Wilayah AWS atau Local Zones. Zona Lokal adalah perpanjangan dari sebuah Wilayah AWS. Ini memungkinkan Anda menempatkan sumber daya komputasi lebih dekat ke pengguna untuk menyediakan gameplay latensi rendah. Untuk informasi selengkapnya, lihat [Zona Lokal AWS](#). Kode lokasi untuk Zona Lokal adalah kode Wilayah induknya diikuti oleh pengidentifikasi lokasi fisik. Misalnya, kode untuk Zona Lokal Los Angeles adalah `us-west-2-lax-1`.

Diagram berikut menggambarkan armada multi-lokasi dengan sumber daya di dua Wilayah AWS dan satu Zona Lokal. Wilayah asal armada adalah `us-west-2`, dan memiliki dua lokasi terpencil: `us-east-2` Wilayah dan Zona `us-west-2-lax-1` Lokal.



Selain sumber daya armada, hosting yang dikelola dengan juga Amazon GameLift Servers dapat menggunakan jenis sumber daya berikut. Anda membuat sumber daya ini secara spesifik Wilayah AWS yang mendukung jenis sumber daya.

- **Build** — Ini adalah server game yang akan di-host dengan EC2 armada terkelola. Buat sumber daya build di wilayah yang sama dengan armada yang akan digunakan.
- **Script** — Ini adalah skrip konfigurasi untuk hosting game dengan Amazon GameLift ServersRealtime. Buat sumber daya skrip di Wilayah yang sama dengan armada yang akan digunakan.
- **Definisi grup kontainer dan gambar kontainer** — Ini adalah konfigurasi untuk menjalankan kontainer pada armada kontainer terkelola. Ini mengidentifikasi satu atau lebih gambar kontainer dengan perangkat lunak untuk menyebarkan ke armada kontainer. Buat definisi grup kontainer dan semua gambar kontainer (yang disimpan dalam repositori Amazon Elastic Container Registry) di Wilayah yang sama dengan armada yang akan digunakan.
- **Antrian sesi permainan** — Sumber daya ini memproses permintaan untuk sesi permainan dan memulai sesi permainan baru. Pemrosesan berlangsung di Wilayah AWS tempat antrian berada. Untuk mengurangi latensi dalam proses penempatan sesi permainan, buat antrian secara geografis di dekat pemain yang akan menggunakannya.

Lokasi untuk Amazon GameLift Servers Di Mana Saja

Armada Amazon GameLift Servers Anywhere adalah kumpulan perangkat keras hosting yang Anda sediakan. Anda mengelola semua aktivitas di sumber daya hosting Anda, termasuk menerapkan

perangkat lunak server game, menjaganya agar tetap diperbarui, dan memulai proses server. Anda membuat armada Anywhere untuk menghubungkan Amazon GameLift Servers layanan dengan sumber daya hosting yang dikelola sendiri. Amazon GameLift Servers mengelola penempatan sesi game - memproses permintaan bergabung pemain, menemukan sumber daya hosting yang tersedia, memulai sesi game baru, dan menyediakan informasi koneksi kepada klien game. Anda dapat membuat armada Anywhere di salah satu Wilayah AWS yang mendukung mereka.

Anda menambahkan instance perangkat keras hosting ke armada Anywhere dengan mendaftarkannya. Setiap instance terdaftar harus memiliki lokasi khusus yang terkait dengannya. Lokasi kustom tidak terkait dengan Wilayah AWS atau Local Zones. Mereka digunakan untuk mewakili lokasi fisik perangkat keras.

Untuk informasi selengkapnya tentang membuat armada Anywhere dan menguji integrasi server game Anda, lihat [Buat sebuah Amazon GameLift Servers Armada di mana saja](#) dan [Siapkan pengujian lokal dengan Amazon GameLift Servers Anywhere](#).

Lokasi untuk Amazon GameLift ServersFlexMatch

FlexMatch sumber daya digunakan untuk memproses permintaan pemain untuk perjodohan. Mereka termasuk sumber daya konfigurasi perjodohan dan sumber daya set aturan. Pemrosesan berlangsung di Wilayah AWS tempat FlexMatch sumber daya berada. Untuk mengurangi latensi dalam proses perjodohan, buat sumber daya secara geografis di dekat pemain yang akan menggunakannya. Konfigurasi perjodohan dan aturan yang digunakannya harus ditempatkan di tempat yang sama. Wilayah AWS Anda dapat membuat FlexMatch sumber daya di salah satu Wilayah AWS yang mendukung mereka.

Untuk informasi selengkapnya tentang pengaturan FlexMatch solusi hosting Anda, lihat [panduan Amazon GameLift ServersFlexMatch pengembang](#).

Amazon GameLift Servers di Tiongkok

Saat menggunakan Amazon GameLift Servers sumber daya di Wilayah Tiongkok (Beijing), yang dioperasikan oleh Sinnet, atau Wilayah Tiongkok (Ningxia), yang dioperasikan oleh NWCD, Anda harus memiliki akun terpisah AWS (Tiongkok). Ketahuilah bahwa beberapa fitur tidak tersedia di Wilayah Tiongkok. Untuk informasi selengkapnya tentang penggunaan Amazon GameLift Servers di Wilayah ini, lihat sumber daya berikut:

- [Amazon Web Services di Tiongkok](#)
- [Amazon GameLift Servers](#) (Memulai Amazon Web Services di Tiongkok)

Harga untuk Amazon GameLift Servers

Harga untuk Amazon GameLift Servers bervariasi berdasarkan jenis layanan dan fitur yang Anda gunakan. Untuk diskusi lengkap tentang penetapan harga, termasuk skenario contoh, lihat [Amazon GameLift Servers Harga](#).

Anda dapat menggunakan Amazon GameLift Servers tanpa dikenakan biaya dengan Tingkat AWS Gratis. Anda dapat menggunakan manfaat Tingkat Gratis jika Anda telah menjadi AWS pelanggan kurang dari 12 bulan dan Anda tetap dalam batas penggunaan tingkat gratis. Untuk informasi selengkapnya, lihat [AWS Tingkat Gratis](#).

Basis biaya untuk masing-masing Amazon GameLift Servers layanan adalah sebagai berikut:

- Amazon GameLift Servers hosting terkelola — Anda membayar berdasarkan (1) penggunaan instans per jam untuk hosting sesi game, dan (2) transfer data untuk lalu lintas antara klien game dan server game yang dihosting. Biaya penggunaan instans bervariasi berdasarkan Wilayah AWS, tipe/ukuran instans, sistem operasi, dan apakah itu instans Spot atau On-Demand.
- Amazon GameLift Servers FlexMatch mandiri — Anda membayar berdasarkan (1) jumlah permintaan pemain untuk perjodohan yang Anda buat, dan (2) waktu pemrosesan yang diperlukan untuk mengevaluasi permintaan pertandingan dan mengusulkan pertandingan.
- Amazon GameLift Servers Hosting di mana saja — Anda membayar berdasarkan (1) jumlah sesi permainan yang Amazon GameLift Servers menempatkan di Anywhere computes, dan (2) jumlah menit koneksi proses server.
- Amazon GameLift Servers FleetiQ — Anda membayar biaya untuk EC2 setiap instans dalam grup server game. Biaya ini berasal dari harga instans per jam dan merupakan tambahan dari biaya EC2 penggunaan. Biaya penggunaan instans bervariasi berdasarkan Wilayah AWS, tipe/ukuran instans, sistem operasi, dan apakah itu instans Spot atau On-Demand.

Topik

- [Menghasilkan Amazon GameLift Servers perkiraan harga](#)
- [Kelola Anda Amazon GameLift Servers biaya hosting](#)

Menghasilkan Amazon GameLift Servers perkiraan harga

Dengan AWS Kalkulator Harga, Anda dapat [membuat perkiraan harga untuk Amazon GameLift Servers](#). Anda tidak perlu pengetahuan mendalam tentang penggunaan AWS kalkulator. Akun AWS

AWS Kalkulator Harga kalkulator memandu Anda melalui keputusan yang memengaruhi biaya layanan untuk memberi Anda gambaran tentang berapa banyak Amazon GameLift Servers Mungkin biaya untuk proyek game Anda. Jika Anda belum yakin bagaimana Anda berencana untuk menggunakan Amazon GameLift Servers, kemudian gunakan nilai default untuk menghasilkan perkiraan. Saat merencanakan penggunaan produksi, kalkulator dapat membantu Anda menguji skenario potensial dan menghasilkan perkiraan yang lebih akurat.

Anda dapat menggunakan AWS Kalkulator Harga untuk menghasilkan perkiraan untuk hal-hal berikut Amazon GameLift Servers opsi hosting:

- [Estimasi Amazon GameLift Servers hosting terkelola](#)

Estimasi Amazon GameLift Servers hosting terkelola

Opsi ini memberikan perkiraan biaya untuk hosting game Anda Amazon GameLift Servers server terkelola, termasuk biaya untuk penggunaan instance server dan transfer data. Dengan Amazon GameLift Servers hosting terkelola, tidak ada biaya tambahan untuk FlexMatch perbandingan.

Jika Anda menghosting atau berencana untuk meng-host server game di lebih dari satu AWS Wilayah atau di lebih dari satu jenis instans, buat perkiraan untuk setiap Wilayah dan jenis instans.

Amazon GameLift Servers Instans

Bagian ini membantu Anda memperkirakan jenis dan jumlah sumber daya komputasi yang Anda butuhkan untuk menyelenggarakan sesi permainan untuk pemain Anda. Amazon GameLift Servers menggunakan [instans Amazon Elastic Compute Cloud \(Amazon EC2\)](#) untuk mengelola server game. Masuk Amazon GameLift Servers, Anda menerapkan armada instance dengan jenis instans dan sistem operasi tertentu. Jika Anda memiliki atau berencana untuk memiliki beberapa armada, buat perkiraan untuk setiap armada.

Untuk memulai, buka [Configure Amazon GameLift Servers halaman](#) dari AWS Kalkulator Harga. Tambahkan Deskripsi, pilih Wilayah, lalu pilih Estimasi Amazon GameLift Servers hosting (Instance +Transfer Data Keluar). Pada Amazon GameLift Servers contoh, lengkapi bidang berikut:

- Puncak pemain bersamaan (puncak CCU)

Ini adalah jumlah maksimum pemain yang dapat terhubung ke server game Anda secara bersamaan. Bidang ini menunjukkan berapa banyak kapasitas hosting Amazon GameLift Servers perlu memenuhi permintaan pemain puncak. Masukkan jumlah puncak harian pemain yang Anda harapkan untuk menjadi tuan rumah menggunakan instance di AWS Wilayah pilihan Anda.

Misalnya, jika Anda ingin membiarkan 1.000 pemain terhubung ke game Anda pada satu waktu, pertahankan nilai default **1000**.

- CCU rata-rata per jam sebagai persentase dari CCU harian puncak

Ini adalah jumlah rata-rata pemain bersamaan per jam selama periode 24 jam. Kami menggunakan nilai ini untuk memperkirakan jumlah kapasitas hosting berkelanjutan yang Amazon GameLift Servers perlu dipertahankan untuk pemain Anda. Jika Anda tidak yakin berapa nilai persentase yang akan digunakan, pertahankan nilai default **50** persen. Untuk game dengan permintaan pemain yang stabil, kami sarankan memasukkan nilai **70** persen.

Misalnya, jika game Anda memiliki CCU rata-rata per jam 6.000 dan CCU puncak 10.000, maka masukkan nilai persen. **60**

- Sesi permainan per instance

Ini adalah jumlah sesi permainan yang dapat dihosting oleh setiap instance server game Anda secara bersamaan. Faktor-faktor yang dapat memengaruhi jumlah ini termasuk persyaratan sumber daya server game Anda, jumlah pemain yang akan dihosting di setiap sesi permainan, dan ekspektasi kinerja pemain. Jika Anda tahu jumlah sesi permainan bersamaan untuk game Anda, maka masukkan nilai itu. Atau, pertahankan nilai default dari **20**.

- Pemain per sesi game

Ini adalah jumlah rata-rata pemain yang terhubung ke sesi permainan, seperti yang didefinisikan dalam desain game Anda. Jika Anda memiliki mode permainan dengan jumlah pemain yang berbeda, perkirakan jumlah rata-rata pemain per sesi game di seluruh game Anda. Nilai default-nya adalah **8**.

- Contoh buffer idle%

Ini adalah persentase kapasitas hosting yang tidak terpakai untuk mempertahankan cadangan untuk menangani lonjakan permintaan pemain yang tiba-tiba. Ukuran buffer adalah persentase dari jumlah total instance dalam armada. Nilai defaultnya adalah **10** persen.

Misalnya, dengan buffer idle 20 persen, armada yang mendukung pemain dengan 100 instans aktif mempertahankan 20 instans idle.

- Contoh spot%

Amazon GameLift Servers armada dapat menggunakan kombinasi Instans Sesuai Permintaan dan Instans Spot. Sementara Instans Sesuai Permintaan menawarkan ketersediaan yang lebih

andal, Instans Spot menawarkan alternatif yang sangat hemat biaya. Kami merekomendasikan penggunaan kombinasi untuk mengoptimalkan penghematan biaya dan ketersediaan. Untuk informasi tentang bagaimana Amazon GameLift Servers menggunakan Instans Spot, lihat [Instans Sesuai Permintaan versus Instans Spot](#).

Untuk bidang ini, masukkan persentase Instans Spot yang akan dipertahankan dalam armada. Kami merekomendasikan persentase Instans Spot antara 50 dan 85 persen. Nilai defaultnya adalah **50** persen.

Misalnya, jika Anda menyebarkan armada dengan 100 instance dan menentukan **40** persen, Amazon GameLift Servers berfungsi untuk mempertahankan 60 Instans Sesuai Permintaan dan 40 Instans Spot.

- Jenis instans

Amazon GameLift Servers armada dapat menggunakan berbagai jenis EC2 instans Amazon yang bervariasi dalam daya komputasi, memori, penyimpanan, dan kemampuan jaringan. Saat Anda mengonfigurasi Amazon GameLift Servers armada, pilih jenis instance yang paling sesuai dengan kebutuhan game Anda. Untuk informasi tentang memilih jenis instans dengan Amazon GameLift Servers, lihat [Pilih sumber daya komputasi untuk armada terkelola](#).

Jika Anda tahu jenis instans yang Anda gunakan atau rencanakan untuk digunakan di Amazon GameLift Servers armada, pilih jenis itu. Jika Anda tidak yakin jenis apa yang harus dipilih, pertimbangkan untuk memilih c5.large. Ini adalah tipe ketersediaan tinggi dengan ukuran dan kemampuan rata-rata.

- Sistem operasi

Bidang ini menentukan sistem operasi yang dijalankan server game Anda—baik Linux atau Windows. Nilai defaultnya adalah Linux.

Transfer data keluar (DTO)

Bagian ini membantu Anda memperkirakan biaya lalu lintas antara klien game Anda dan server game. Biaya transfer data hanya berlaku untuk lalu lintas keluar. Transfer data masuk tidak memiliki biaya.

Pada [Konfigurasi Amazon GameLift Servers halaman](#) AWS Kalkulator Harga, memperluas Transfer data keluar (DTO), dan kemudian lengkapi bidang berikut:

- Jenis estimasi DTO

Anda dapat memilih untuk memperkirakan DTO dalam salah satu dari dua cara berikut, tergantung pada bagaimana Anda melacak transfer data untuk game Anda.

- Per bulan (dalam GB) — Jika Anda melacak lalu lintas bulanan untuk server game Anda, pilih jenis ini.
- Per pemain — Jika Anda melacak transfer data berdasarkan pemain, pilih jenis ini. Ini adalah tipe default.

Di bidang berikut, Anda memperkirakan DTO per pemain berdasarkan jumlah jam pemain yang Anda hitung di bagian sebelumnya.

- DTO per bulan (dalam GB)

Jika Anda memilih jenis estimasi DTO Per bulan (dalam GB), masukkan perkiraan penggunaan DTO bulanan Anda dalam GB dari setiap instans, per Wilayah.

- DTO per pemain

Jika Anda memilih jenis perkiraan DTO Per pemain, maka masukkan perkiraan penggunaan DTO game Anda per pemain dalam KB/detik. Nilai default-nya adalah **4**.

Setelah selesai mengonfigurasi Amazon GameLift Servers estimasi harga, pilih Tambahkan ke perkiraan saya. Untuk informasi selengkapnya tentang membuat dan mengelola estimasi AWS Kalkulator Harga, lihat [Membuat estimasi, mengonfigurasi layanan, dan menambahkan layanan lainnya](#) di Panduan AWS Kalkulator Harga Pengguna.

Kelola Anda Amazon GameLift Servers biaya hosting

AWS Tagihan Anda mencerminkan biaya hosting game Anda. Anda dapat melihat perkiraan biaya untuk bulan berjalan, dan biaya akhir untuk bulan sebelumnya di konsol Penagihan di <https://console.aws.amazon.com/costmanagement/>. Untuk informasi selengkapnya tentang alat dan sumber daya untuk membantu Anda mengelola AWS biaya, lihat [Panduan AWS Billing Pengguna](#). Panduan ini dapat membantu Anda meninjau konsumsi sumber daya Anda, menetapkan penggunaan masa depan, dan menentukan kebutuhan penskalaan Anda.

Pertimbangkan tips ini untuk membantu Anda mengelola biaya Amazon GameLift Servers layanan.

Buat peringatan penagihan untuk memantau penggunaan

Siapkan peringatan penggunaan Tingkat AWS Gratis untuk memberi tahu Anda saat penggunaan Anda mendekati atau melebihi batas Tingkat Gratis Amazon GameLift Servers dan lainnya Layanan AWS. Anda dapat mengonfigurasi peringatan untuk mengambil tindakan berdasarkan tingkat penggunaan Anda. Misalnya, Anda dapat secara otomatis mengatur anggaran Anda ke nol ketika Anda mencapai batas Tingkat Gratis.

Anda juga dapat mengatur peringatan CloudWatch penagihan Amazon untuk mendapatkan notifikasi saat penggunaan mencapai ambang batas khusus.

Untuk informasi selengkapnya, lihat topik ini di Panduan AWS Billing Pengguna:

- [Melacak penggunaan Tingkat AWS Gratis Anda](#)
- [Preferensi peringatan penagihan](#)

Lacak biaya per Amazon GameLift Servers armada

Gunakan tag alokasi AWS biaya untuk mengatur dan melacak biaya hosting game Anda berdasarkan Amazon GameLift Servers EC2 Armada Amazon dan sumber daya lainnya. Dengan menandai armada Anda, baik secara individu maupun grup, Anda dapat membuat laporan alokasi biaya yang mengkategorikan biaya berdasarkan tag yang ditetapkan. Anda dapat menggunakan jenis laporan ini untuk mengidentifikasi bagaimana armada berkontribusi terhadap biaya hosting Anda. Anda juga dapat menggunakan tag untuk memfilter tampilan AWS Cost Explorer.

Untuk informasi lebih lanjut, lihat topik-topik ini:

- [Menggunakan tag alokasi AWS biaya](#), AWS Billing Panduan Pengguna
- [Menganalisis biaya Anda dengan AWS Cost Explorer](#), Panduan AWS Cost Management Pengguna

Atur kapasitas armada terkelola ke nol

Armada yang dikelola dapat terus mengeluarkan biaya bahkan ketika mereka tidak digunakan untuk menyelenggarakan sesi permainan. Untuk menghindari biaya yang tidak perlu, [turunkan armada Anda](#) ke nol saat tidak digunakan. Jika Anda menggunakan penskalaan otomatis, hentikan aktivitas ini dan setel kapasitas armada secara manual.

Memulai dengan Amazon GameLift Servers

Manfaatkan sumber daya untuk memulai ini untuk mempelajari lebih lanjut tentang Amazon GameLift Servers layanan ini dan bagaimana Anda dapat mulai mengembangkan solusi hosting khusus untuk game multipemain berbasis sesi Anda.

Sebelum Anda mulai

- Buat Akun AWS (atau tentukan yang sudah ada) untuk digunakan. Amazon GameLift Servers
- Siapkan pengguna dengan izin untuk Amazon GameLift Servers dan AWS layanan terkait.
- Pilih sebuah Wilayah AWS untuk bekerja di. Untuk pengembangan, pilih Wilayah yang dekat dengan lokasi Anda. Anda dapat mengubah Wilayah kapan saja.

[Mengatur sebuah Akun AWS](#)

Opsi orientasi cepat

Cobalah alat mulai cepat ini untuk mendapatkan solusi hosting dasar dan berjalan cepat dengan pengembangan yang efisien. Alat-alat ini ideal untuk bukti konsep dan prototipe, atau menggunakannya untuk membangun lingkungan pengujian untuk pengembangan game iteratif yang cepat. Setelah menggunakan alat ini untuk menyebarkan server game untuk hosting, Anda dapat menggunakan Amazon GameLift Servers konsol dan alat API untuk memantau kinerja armada, mengelola sesi game, dan menganalisis metrik.

- [Pembungkus server game untuk Amazon GameLift Servers](#) - Alat ini dan [tutorial orientasi](#) untuknya adalah cara tercepat dan termudah untuk membuat server game Anda di-host dan menjalankan sesi game Amazon GameLift Servers tanpa perlu perubahan kode. Pembungkus server game Readme menyediakan instruksi baris perintah untuk semua jenis armada dan sorotan tutorial menggunakan konsol dan jenis EC2 armada terkelola sehingga Anda dapat segera memulai. Ikuti petunjuk untuk menyiapkan manajemen sesi game dan penyebaran server game yang efisien.

Saat Anda siap membangun solusi hosting game khusus, beralihlah ke salah satu opsi pengembangan khusus dengan integrasi penuh dengan SDK server untuk Amazon GameLift Servers. Jika game Anda tidak memerlukan solusi hosting khusus, Anda dapat terus menggunakan pembungkus server game untuk menyebarkan dan meng-host server game dalam produksi.

- [Amazon GameLift Serversplugin untuk Unreal Engine atau Unity](#) - Plugin memberi Anda alur kerja GUI dan aset sampel untuk memandu Anda melalui langkah-langkah awal dan menyebarkan server game Anda dengan solusi hosting dasar. Gunakan plugin untuk menyiapkan hosting dengan armada Anywhere yang dikelola sendiri, atau menyebarkan armada atau armada kontainer berbasis cloud yang dikelola EC2 . Ketika Anda siap untuk mengembangkan solusi hosting khusus, Anda dapat membangun solusi yang dibuat plugin Anda.
- [Starter kit untuk kontainer Amazon GameLift Servers terkelola](#) - Kit ini merampingkan tugas untuk mengintegrasikan server game, menyiapkan gambar kontainer server game, dan menyebarkan armada kontainer untuk hosting. Untuk integrasi, kit menambahkan fitur manajemen sesi permainan penting ke server game Anda. Kit ini menggunakan templat yang telah dikonfigurasi sebelumnya untuk membangun armada kontainer dan pipa penyebaran otomatis untuk server game. Saat Anda siap menambahkan fitur manajemen sesi game lengkap, ikuti salah satu peta jalan pengembangan khusus untuk mengintegrasikan SDK server. Amazon GameLift Servers

Opsi pengembangan khusus

Ikuti salah satu peta jalan pengembangan ini untuk mulai membangun solusi hosting khusus berfitur lengkap untuk game Anda. Peta jalan memberikan panduan terperinci tentang cara membuat, menguji, dan menyesuaikan setiap komponen dalam solusi hosting Anda.

- [Peta jalan pengembangan untuk hosting dengan Amazon GameLift Servers terkelola EC2](#)
- [Peta jalan pengembangan untuk hosting dengan kontainer Amazon GameLift Servers terkelola](#)
- [Peta jalan pengembangan untuk hosting dengan Amazon GameLift Servers Anywhere](#)
- [Peta jalan pengembangan untuk hosting hybrid dengan Amazon GameLift Servers](#)

Amazon GameLift Servers contoh

Jika Anda mempertimbangkan Amazon GameLift Servers untuk menggunakan untuk mengelola server game kustom Anda, atau Anda tertarik untuk memanfaatkannya Amazon GameLift ServersRealtime, kami sarankan Anda mencoba contoh berikut sebelum Anda menggunakan layanan untuk game Anda sendiri. Contoh server game khusus memberi Anda pengalaman dengan hosting game di Amazon GameLift Servers konsol. Amazon GameLift ServersRealtimeContoh menunjukkan kepada Anda bagaimana mempersiapkan game untuk hosting menggunakan Realtime server.

Contoh server game kustom

Contoh ini menunjukkan proses penyebaran server game sampel ke EC2 armada Amazon GameLift Servers terkelola untuk hosting. Gunakan klien game sampel untuk terhubung ke sesi permainan langsung. Anda dapat merasakan cara Amazon GameLift Servers menggunakan `tools`, termasuk konsol dan AWS CLI, untuk memantau kinerja dan penggunaan hosting armada.

Contoh memandu Anda melalui langkah-langkah berikut:

- Unggah contoh pembuatan server game.
- Buat armada untuk menjalankan build server game.
- Dapatkan contoh klien game dan gunakan untuk terhubung ke server game dan bergabung dengan sesi permainan.
- Tinjau metrik armada dan sesi game.

Mulai beberapa klien game dan mainkan game untuk menghasilkan data hosting. Gunakan Amazon GameLift Servers konsol untuk melihat sumber daya hosting, melacak metrik, dan menjelajahi opsi untuk menskalakan kapasitas hosting armada.

Untuk memulai, masuk ke [Amazon GameLift Servers konsol](#). Di navigasi sisi kiri, buka Sumber Daya, Coba contoh permainan.

Amazon GameLift Servers Realtime contoh

Contoh ini adalah tutorial lengkap yang memandu Anda melalui cara menyebarkan contoh game multipemain, Mega Frog Race, dengan Amazon GameLift Servers Realtime Tutorial ini mencakup cara mengintegrasikan klien game Anda dengan Realtime SDK dan menerapkan solusi hosting lengkap dengan Realtime server pada armada terkelola EC2.

Untuk tutorial langsung, lihat [Membuat Server untuk Game Seluler Multiplayer dengan Hanya Beberapa Baris JavaScript di blog AWS for Games](#). Untuk kode sumber Mega Frog Race, lihat [GitHub repositori](#).

Kode sumber mencakup bagian-bagian berikut:

- Klien game — Kode sumber untuk klien game C++, dibuat di Unity. Klien game mendapatkan informasi koneksi sesi game, terhubung ke server, dan bertukar pembaruan dengan pemain lain.
- Layanan backend — Kode sumber untuk AWS Lambda fungsi yang mengelola panggilan langsung ke API layanan untuk Amazon GameLift Servers

- Realscript — File skrip sumber yang mengkonfigurasi armada Realtime server untuk game. Skrip ini mencakup konfigurasi minimum yang diperlukan untuk setiap Realtime server untuk berkomunikasi Amazon GameLift Servers dan menyelenggarakan sesi permainan.

Setelah Anda mengatur contoh game untuk hosting, gunakan sebagai titik awal untuk bereksperimen dengan Amazon GameLift Servers fitur lain seperti FlexMatch.

Mengatur sebuah Akun AWS

Untuk mulai menggunakan Amazon GameLift Servers, buat dan atur Akun AWS. Tidak ada biaya untuk membuat Akun AWS. Bagian ini memandu Anda dalam membuat akun, menyiapkan pengguna, dan mengonfigurasi izin.

Topik

- [Mendaftar untuk Akun AWS](#)
- [Buat pengguna dengan akses administratif](#)
- [Tetapkan izin pengguna untuk Amazon GameLift Servers](#)
- [Menyiapkan akses terprogram untuk pengguna](#)
- [Siapkan akses terprogram untuk game Anda](#)
- [Contoh izin IAM untuk Amazon GameLift Servers](#)
- [Menyiapkan peran layanan IAM untuk Amazon GameLift Servers](#)

Mendaftar untuk Akun AWS

Jika Anda tidak memiliki Akun AWS, selesaikan langkah-langkah berikut untuk membuatnya.

Untuk mendaftar untuk Akun AWS

1. Buka <https://portal.aws.amazon.com/billing/pendaftaran>.
2. Ikuti petunjuk online.

Bagian dari prosedur pendaftaran melibatkan menerima panggilan telepon atau pesan teks dan memasukkan kode verifikasi pada keypad telepon.

Saat Anda mendaftar untuk sebuah Akun AWS, sebuah Pengguna root akun AWS dibuat. Pengguna root memiliki akses ke semua Layanan AWS dan sumber daya di akun. Sebagai

praktik keamanan terbaik, tetapkan akses administratif ke pengguna, dan gunakan hanya pengguna root untuk melakukan [tugas yang memerlukan akses pengguna root](#).

AWS mengirim Anda email konfirmasi setelah proses pendaftaran selesai. Kapan saja, Anda dapat melihat aktivitas akun Anda saat ini dan mengelola akun Anda dengan masuk <https://aws.amazon.com/ke/> dan memilih Akun Saya.

Buat pengguna dengan akses administratif

Setelah Anda mendaftar Akun AWS, amankan Pengguna root akun AWS, aktifkan AWS IAM Identity Center, dan buat pengguna administratif sehingga Anda tidak menggunakan pengguna root untuk tugas sehari-hari.

Amankan Anda Pengguna root akun AWS

1. Masuk ke [AWS Management Console](#) sebagai pemilik akun dengan memilih pengguna Root dan memasukkan alamat Akun AWS email Anda. Di laman berikutnya, masukkan kata sandi.

Untuk bantuan masuk dengan menggunakan pengguna root, lihat [Masuk sebagai pengguna root](#) di AWS Sign-In Panduan Pengguna.

2. Mengaktifkan autentikasi multi-faktor (MFA) untuk pengguna root Anda.

Untuk petunjuk, lihat [Mengaktifkan perangkat MFA virtual untuk pengguna Akun AWS root \(konsol\) Anda](#) di Panduan Pengguna IAM.

Buat pengguna dengan akses administratif

1. Aktifkan Pusat Identitas IAM.

Untuk mendapatkan petunjuk, silakan lihat [Mengaktifkan AWS IAM Identity Center](#) di Panduan Pengguna AWS IAM Identity Center .

2. Di Pusat Identitas IAM, berikan akses administratif ke pengguna.

Untuk tutorial tentang menggunakan Direktori Pusat Identitas IAM sebagai sumber identitas Anda, lihat [Mengkonfigurasi akses pengguna dengan default Direktori Pusat Identitas IAM](#) di Panduan AWS IAM Identity Center Pengguna.

Masuk sebagai pengguna dengan akses administratif

- Untuk masuk dengan pengguna Pusat Identitas IAM, gunakan URL masuk yang dikirim ke alamat email saat Anda membuat pengguna Pusat Identitas IAM.

Untuk bantuan masuk menggunakan pengguna Pusat Identitas IAM, lihat [Masuk ke portal AWS akses](#) di Panduan AWS Sign-In Pengguna.

Tetapkan akses ke pengguna tambahan

1. Di Pusat Identitas IAM, buat set izin yang mengikuti praktik terbaik menerapkan izin hak istimewa paling sedikit.

Untuk petunjuknya, lihat [Membuat set izin](#) di Panduan AWS IAM Identity Center Pengguna.

2. Tetapkan pengguna ke grup, lalu tetapkan akses masuk tunggal ke grup.

Untuk petunjuk, lihat [Menambahkan grup](#) di Panduan AWS IAM Identity Center Pengguna.

Tetapkan izin pengguna untuk Amazon GameLift Servers

Buat pengguna tambahan atau perluas izin akses ke pengguna yang ada sesuai kebutuhan untuk Amazon GameLift Servers sumber daya Anda. Sebagai praktik terbaik (Praktik [terbaik keamanan di IAM](#)), terapkan izin hak istimewa paling sedikit untuk semua pengguna. Untuk panduan tentang sintaks izin, lihat [Contoh izin IAM untuk Amazon GameLift Servers](#)

Gunakan petunjuk berikut untuk menetapkan izin pengguna berdasarkan cara Anda mengelola pengguna di AWS akun Anda.

Untuk memberikan akses dan menambahkan izin bagi pengguna, grup, atau peran Anda:

- Pengguna dan grup di AWS IAM Identity Center:

Buat rangkaian izin. Ikuti instruksi di [Buat rangkaian izin](#) di Panduan Pengguna AWS IAM Identity Center .

- Pengguna yang dikelola di IAM melalui penyedia identitas:

Buat peran untuk federasi identitas. Ikuti instruksi dalam [Buat peran untuk penyedia identitas pihak ketiga \(federasi\)](#) dalam Panduan Pengguna IAM.

- Pengguna IAM:

- Buat peran yang dapat diambil pengguna Anda. Ikuti instruksi dalam [Buat peran untuk pengguna IAM](#) dalam Panduan Pengguna IAM.
- (Tidak disarankan) Lampirkan kebijakan langsung ke pengguna atau tambahkan pengguna ke grup pengguna. Ikuti petunjuk dalam [Menambahkan izin ke pengguna \(konsol\)](#) dalam Panduan Pengguna IAM.

Saat bekerja dengan pengguna IAM, sebagai praktik terbaik selalu lampirkan izin ke peran atau grup pengguna, bukan pengguna individu.

Menyiapkan akses terprogram untuk pengguna

Pengguna membutuhkan akses terprogram jika mereka ingin berinteraksi dengan AWS luar. AWS Management Console Cara untuk memberikan akses terprogram tergantung pada jenis pengguna yang mengakses AWS.

Untuk memberi pengguna akses programatis, pilih salah satu opsi berikut.

Pengguna mana yang membutuhkan akses programatis?	Untuk	Oleh
Identitas tenaga kerja (Pengguna yang dikelola di Pusat Identitas IAM)	Gunakan kredensi sementara untuk menandatangani permintaan terprogram ke AWS CLI,, AWS SDKs atau. AWS APIs	Mengikuti petunjuk untuk antarmuka yang ingin Anda gunakan. • Untuk AWS CLI, lihat Mengkonfigurasi yang akan AWS CLI digunakan AWS IAM Identity Center dalam Panduan AWS Command Line Interface Pengguna. • Untuk AWS SDKs, alat, dan AWS APIs, lihat Autentikasi Pusat Identitas IAM di Panduan Referensi Alat AWS SDKs dan Alat.

Pengguna mana yang membutuhkan akses programatis?	Untuk	Oleh
IAM	Gunakan kredensi sementara untuk menandatangani permintaan terprogram ke AWS CLI,, AWS SDKs atau. AWS APIs	Mengikuti petunjuk dalam Menggunakan kredensial sementara dengan AWS sumber daya di Panduan Pengguna IAM.
IAM	(Tidak direkomendasikan) Gunakan kredensi jangka panjang untuk menandatangani permintaan terprogram ke AWS CLI,, AWS SDKs atau. AWS APIs	Mengikuti petunjuk untuk antarmuka yang ingin Anda gunakan. • Untuk mengetahui AWS CLI, lihat Mengautentikasi menggunakan kredensial pengguna IAM di Panduan Pengguna. AWS Command Line Interface • Untuk AWS SDKs dan alat, lihat Mengautentikasi menggunakan kredensial jangka panjang di Panduan Referensi Alat AWS SDKs dan Alat. • Untuk AWS APIs, lihat Mengelola kunci akses untuk pengguna IAM di Panduan Pengguna IAM.

Jika Anda menggunakan kunci akses, lihat [Praktik terbaik untuk mengelola kunci AWS akses](#).

Siapkan akses terprogram untuk game Anda

Sebagian besar game menggunakan layanan backend untuk berkomunikasi dengan Amazon GameLift Servers menggunakan. AWS SDKs Gunakan layanan backend (bertindak untuk klien

game) untuk meminta sesi permainan, menempatkan pemain ke dalam game, dan tugas lainnya. Layanan ini memerlukan akses terprogram dan kredensi keamanan untuk mengautentikasi panggilan ke API layanan. Amazon GameLift Servers

Untuk Amazon GameLift Servers, Anda mengelola akses ini dengan membuat pengguna pemain di AWS Identity and Access Management (IAM). Kelola izin pengguna pemain melalui salah satu opsi berikut:

- Buat peran IAM dengan izin pengguna pemain dan izinkan pengguna pemain untuk mengambil peran saat diperlukan. Layanan backend harus menyertakan kode untuk mengambil peran ini sebelum membuat permintaan. Amazon GameLift Servers Sesuai dengan praktik terbaik keamanan, peran menyediakan akses sementara yang terbatas. Anda dapat menggunakan peran untuk beban kerja yang berjalan pada AWS sumber daya ([peran IAM](#)) atau di luar (Peran [IAM](#) Di AWS Mana Saja).
- Buat grup pengguna IAM dengan izin pengguna pemain dan tambahkan pengguna pemain Anda ke grup. Opsi ini memberikan kredensi jangka panjang pengguna pemain Anda, yang harus disimpan dan digunakan oleh layanan backend saat berkomunikasi. Amazon GameLift Servers

Untuk sintaks kebijakan izin, lihat. [Contoh izin pengguna pemain](#)

Untuk informasi selengkapnya tentang mengelola izin untuk digunakan oleh beban kerja, lihat [Identitas IAM: Kredensial sementara](#) di IAM.

Contoh izin IAM untuk Amazon GameLift Servers

Gunakan sintaks dalam contoh ini untuk menetapkan izin AWS Identity and Access Management (IAM) bagi pengguna yang membutuhkan akses ke sumber daya. Amazon GameLift Servers Untuk informasi selengkapnya tentang mengelola izin pengguna, lihat [Tetapkan izin pengguna untuk Amazon GameLift Servers](#). Saat mengelola izin untuk pengguna di luar Pusat Identitas IAM, sebagai praktik terbaik selalu lampirkan izin ke peran IAM atau grup pengguna, bukan pengguna individu.

Jika Anda menggunakan Amazon GameLift Servers FleetIQ sebagai solusi mandiri, lihat [Mengatur Akun AWS untuk Amazon GameLift Servers FleetIQ Anda](#).

Contoh izin administrasi

Contoh-contoh ini memberikan administrator hosting atau akses yang ditargetkan pengembang untuk mengelola sumber daya hosting Amazon GameLift Servers game.

Example Sintaks untuk izin sumber daya akses Amazon GameLift Servers penuh

Contoh berikut memperluas akses penuh ke semua Amazon GameLift Servers sumber daya.

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

Example Sintaks untuk izin Amazon GameLift Servers sumber daya dengan dukungan untuk Wilayah yang tidak diaktifkan secara default

Contoh berikut memperluas akses ke semua Amazon GameLift Servers sumber daya dan AWS Wilayah yang tidak diaktifkan secara default. Untuk informasi selengkapnya tentang Wilayah yang tidak diaktifkan secara default dan cara mengaktifkannya, lihat [Mengelola Wilayah AWS](#) di Referensi Umum AWS.

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

Example Sintaks untuk Amazon GameLift Servers sumber daya untuk mengakses gambar kontainer di Amazon ECR

Contoh berikut memperluas akses ke tindakan Amazon Elastic Container Registry (Amazon ECR) Registry (Amazon ECR) yang diperlukan Amazon GameLift Servers pengguna saat bekerja dengan armada kontainer terkelola.

```
{
```

```

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

```

Example Sintaks untuk Amazon GameLift Servers sumber daya dan izin **PassRole**

Contoh berikut memperluas akses ke semua Amazon GameLift Servers sumber daya dan memungkinkan pengguna untuk meneruskan peran layanan IAM ke. Amazon GameLift Servers Peran layanan memberikan kemampuan Amazon GameLift Servers terbatas untuk mengakses sumber daya dan layanan lain atas nama Anda, seperti yang dijelaskan dalam [Menyiapkan peran layanan IAM untuk Amazon GameLift Servers](#). Misalnya, saat menanggapi CreateBuild permintaan, Amazon GameLift Servers memerlukan akses ke file build Anda di bucket Amazon S3. Untuk informasi selengkapnya tentang PassRole tindakan, lihat [IAM: Meneruskan peran IAM ke AWS layanan tertentu di Panduan Pengguna IAM](#).

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "gamelift:*",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:PassRole",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "iam:PassedToService": "gamelift.amazonaws.com"
        }
      }
    }
  ]
}

```

```
}
```

Contoh izin pengguna pemain

Contoh-contoh ini memungkinkan layanan backend atau entitas lain untuk melakukan panggilan API ke API. Amazon GameLift Servers Mereka mencakup skenario umum untuk mengelola sesi permainan, sesi pemain, dan perjodohan. Untuk detail selengkapnya, lihat [Siapkan akses terprogram untuk game Anda](#).

Example Sintaks untuk izin penempatan sesi game

Contoh berikut memperluas akses ke antrian penempatan sesi permainan Amazon GameLift Servers APIs yang menggunakan untuk membuat sesi permainan dan mengelola sesi pemain.

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

Example Sintaks untuk izin perjodohan

Contoh berikut memperluas akses ke Amazon GameLift Servers APIs yang mengelola kegiatan FlexMatch perjodohan. FlexMatch mencocokkan pemain untuk sesi permainan baru atau yang sudah ada dan memulai penempatan sesi permainan untuk game yang diselenggarakan. Amazon GameLift Servers Untuk informasi lebih lanjut tentang FlexMatch, lihat [Apa itu 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 Sintaks untuk izin penempatan sesi permainan manual

Contoh berikut memperluas akses ke Amazon GameLift Servers APIs yang secara manual membuat sesi permainan dan sesi pemain pada armada tertentu. Skenario ini mendukung game yang tidak menggunakan antrian penempatan, seperti game yang memungkinkan pemain bergabung dengan memilih dari daftar sesi permainan yang tersedia (metode 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": "*"
  }
}
```

Menyiapkan peran layanan IAM untuk Amazon GameLift Servers

Beberapa Amazon GameLift Servers fitur mengharuskan Anda untuk memperluas akses terbatas ke AWS sumber daya lain yang Anda miliki. Anda dapat melakukan ini dengan membuat peran AWS

Identity and Access Management (IAM). Sebuah [peran IAM](#) adalah identitas IAM yang dapat Anda buat di akun yang memiliki izin tertentu. Peran IAM mirip dengan pengguna IAM karena merupakan AWS identitas dengan kebijakan izin yang menentukan apa yang dapat dan tidak dapat dilakukan identitas. AWS Namun, alih-alih secara unik terkait dengan satu orang, peran dimaksudkan untuk menjadi dapat diambil oleh siapa pun yang membutuhkannya. Selain itu, peran tidak memiliki kredensial jangka panjang standar seperti kata sandi atau kunci akses yang terkait dengannya. Sebagai gantinya, saat Anda mengambil peran, peran tersebut akan memberikan kredensial keamanan sementara untuk sesi peran.

Topik ini mencakup cara membuat peran yang dapat Anda gunakan dengan armada Amazon GameLift Servers terkelola Anda. Jika Anda menggunakannya Amazon GameLift Servers FleetIQ untuk mengoptimalkan hosting game di instans Amazon Elastic Compute Cloud (Amazon EC2), lihat [Mengatur untuk Anda Akun AWS](#). Amazon GameLift Servers FleetIQ

Dalam prosedur berikut, buat peran dengan kebijakan izin khusus dan kebijakan kepercayaan yang memungkinkan Amazon GameLift Servers untuk mengambil peran tersebut.

Buat peran layanan IAM untuk armada Amazon GameLift Servers terkelola EC2

Langkah 1: Buat kebijakan izin.

Gunakan petunjuk dan contoh di halaman ini untuk membuat kebijakan izin khusus untuk jenis Amazon GameLift Servers armada yang Anda gunakan.

Cara menggunakan editor kebijakan JSON untuk membuat kebijakan

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Pada panel navigasi di sebelah kiri, pilih Kebijakan.

Jika ini pertama kalinya Anda memilih Kebijakan, akan muncul halaman Selamat Datang di Kebijakan Terkelola. Pilih Memulai.

3. Di bagian atas halaman, pilih Buat kebijakan.
4. Di bagian Editor kebijakan, pilih opsi JSON.
5. Masukkan atau tempel dokumen kebijakan JSON. Untuk detail bahasa kebijakan IAM, lihat [Referensi kebijakan JSON IAM](#).
6. Selesaikan peringatan keamanan, kesalahan, atau peringatan umum yang dihasilkan selama [validasi kebijakan](#), lalu pilih Berikutnya.

 Note

Anda dapat beralih antara opsi editor Visual dan JSON kapan saja. Namun, jika Anda melakukan perubahan atau memilih Berikutnya di editor Visual, IAM dapat merestrukturisasi kebijakan Anda untuk mengoptimalkannya bagi editor visual. Untuk informasi selengkapnya, lihat [Restrukturisasi kebijakan](#) dalam Panduan Pengguna IAM.

7. (Opsional) Saat membuat atau mengedit kebijakan AWS Management Console, Anda dapat membuat templat kebijakan JSON atau YAMAL yang dapat Anda gunakan dalam AWS CloudFormation templat.

Untuk melakukan ini, di editor Kebijakan pilih Tindakan, lalu pilih Buat CloudFormation templat. Untuk mempelajari selengkapnya AWS CloudFormation, lihat [referensi jenis AWS Identity and Access Management sumber daya](#) di Panduan AWS CloudFormation Pengguna.

8. Setelah selesai menambahkan izin ke kebijakan, pilih Berikutnya.
9. Pada halaman Tinjau dan buat, masukkan Nama kebijakan dan Deskripsi (opsional) untuk kebijakan yang Anda buat. Tinjau Izin yang ditentukan dalam kebijakan ini untuk melihat izin yang diberikan oleh kebijakan Anda.
10. (Opsional) Tambahkan metadata ke kebijakan dengan melampirkan tanda sebagai pasangan nilai kunci. Untuk informasi selengkapnya tentang penggunaan tag di IAM, lihat [Tag untuk AWS Identity and Access Management sumber daya](#) di Panduan Pengguna IAM.
11. Pilih Buat kebijakan untuk menyimpan kebijakan baru Anda.

Langkah 2: Buat peran yang Amazon GameLift Servers dapat diasumsikan.

Untuk membuat IAM role

1. Di panel navigasi konsol IAM, pilih Peran, dan lalu pilih Buat peran.
2. Pada halaman Pilih entitas tepercaya, pilih opsi Kebijakan kepercayaan kustom. Pilihan ini membuka editor kebijakan kepercayaan kustom.
3. Ganti sintaks JSON default dengan yang berikut ini, lalu pilih Berikutnya untuk melanjutkan.

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

```
    "Effect": "Allow",
    "Principal": {
        "Service": "gamelift.amazonaws.com"
    },
    "Action": "sts:AssumeRole"
  }
]
```

4. Pada halaman Tambahkan izin, cari dan pilih kebijakan izin yang Anda buat di Langkah 1. Pilih Next untuk melanjutkan.
5. Pada halaman Nama, tinjau, dan buat, masukkan nama Peran dan Deskripsi (opsional) untuk peran yang Anda buat. Tinjau entitas Trust dan Izin tambahan.
6. Pilih Buat peran untuk menyimpan peran baru Anda.

Buat peran IAM untuk kontainer Amazon GameLift Servers terkelola

Jika Anda menggunakan kontainer Amazon GameLift Servers terkelola, Anda perlu membuat peran layanan IAM untuk digunakan dengan armada kontainer. Peran ini memberikan izin terbatas yang Amazon GameLift Servers diperlukan untuk mengelola sumber daya armada kontainer Anda dan mengambil tindakan atas nama Anda.

Untuk membuat peran IAM untuk armada kontainer

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi konsol IAM, pilih Peran, dan lalu pilih Buat peran.
3. Pada halaman Pilih entitas tepercaya, pilih AWS layanan dan pilih Kasus penggunaan "GameLift". Pilih Berikutnya
4. Pada Tambahkan izin, pilih kebijakan GameLiftContainerFleetPolicy terkelola. Pilih Berikutnya. Lihat [AWS kebijakan terkelola untuk Amazon GameLift Servers](#) untuk informasi lebih lanjut tentang kebijakan ini.
5. Pada Nama, tinjau, dan buat, masukkan nama peran dan pilih Buat peran untuk menyimpan peran baru.

Sintaks kebijakan izin

- Izin Amazon GameLift Servers untuk mengambil peran layanan

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

- Izin untuk mengakses AWS Wilayah yang tidak diaktifkan secara default

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

Dapatkan alat Amazon GameLift Servers pengembangan

Amazon GameLift Servers menyediakan seperangkat alat lain untuk membantu Anda membangun solusi hosting game untuk game Anda. SDKs SDKs Tambahkan fungsionalitas ke server game, klien game, dan layanan backend yang memungkinkan mereka berinteraksi dengan layanan. Amazon GameLift Servers Untuk informasi terbaru tentang versi dan kompatibilitas Amazon GameLift Servers SDK, lihat [Catatan rilis Amazon GameLift Servers](#).

Untuk server game

Integrasikan dan bangun server game 64-bit Anda dengan server SDK untuk Amazon GameLift Servers. Server game menggunakan SDK server untuk berkomunikasi dengan Amazon GameLift Servers layanan untuk manajemen sesi game, termasuk memulai, memperbarui, dan menghentikan sesi permainan. Untuk bantuan mengintegrasikan SDK server ke dalam proyek game Anda, lihat [Mempersiapkan game untuk Amazon GameLift Servers](#)

Dukungan pengembangan

- Pengembangan OS
 - Windows
 - Linux
- Bahasa pemrograman

[Dapatkan Amazon GameLift Servers SDK](#). Untuk informasi spesifik versi dan petunjuk penginstalan, lihat file readme yang disertakan di setiap paket.

- [Server C++ SDK](#)
 - [Referensi SDK Server](#)
 - [Bagaimana cara mengintegrasikan](#)
- [C # server SDK](#) (Support untuk .NET 4, .NET 6, .NET 8 bervariasi menurut versi, lihat) [Versi SDK](#)
 - [Referensi SDK Server](#)
 - [Bagaimana cara mengintegrasikan](#)
- [Pergi SDK server](#)
 - [Referensi SDK Server](#)
 - [Bagaimana cara mengintegrasikan](#)
- Dukungan mesin game

Plugin berfitur lengkap untuk Amazon GameLift Servers menyertakan alur kerja UI dan aset sampel, serta versi bawaan AWS SDK dan SDK server. Alur kerja memandu Anda melalui cara mengonfigurasi dan menerapkan server game Anda untuk hosting dengan EC2 armada terkelola, armada kontainer terkelola, atau armada Anywhere yang dikelola sendiri. Jika Anda tidak memerlukan alur kerja yang dipandu, Anda bisa mendapatkan SDK server untuk mesin game Anda dari repositori yang sama GitHub .

Jika Anda menggunakan mesin game atau lingkungan pengembangan lain yang tidak didukung plugin, dapatkan SDK server untuk bahasa pemrograman Anda dan tambahkan ke proyek game Anda.

Untuk informasi spesifik versi dan petunjuk penginstalan, lihat file readme yang disertakan di setiap paket.

- [Plugin untuk Unreal Engine](#) - Dibangun untuk digunakan dengan Unreal Engine versi 5.2, 5.3, 5.4, dan 5.5. Periksa readmes khusus versi untuk dukungan Unreal.
 - [Panduan plugin untuk Unreal Engine](#)
 - [Server C ++ \(Tidak nyata\) SDK 5.x untuk -- Tindakan Amazon GameLift Servers](#)
- [Plugin untuk Unity](#) - Dibangun untuk digunakan dengan versi LTS Unity Editor 6.0, 2022.3, atau 2021.3. Ini mendukung profil .NET Framework dan .NET Standard Unity, dengan .NET Standard 2.1 dan .NET 4.x. Periksa readmes khusus versi untuk dukungan Unity.
 - [Panduan plugin untuk Unity](#)
 - [???Referensi SDK server C #](#)
- [Server SDK untuk Unreal](#)
 - [Referensi SDK Server](#)
 - [Integrasikan Amazon GameLift Servers ke dalam proyek Unreal Engine](#)
- [Server SDK for Unity](#)
 - [Referensi SDK Server](#)
 - [Integrasikan Amazon GameLift Servers ke dalam proyek Unity](#)

Dukungan runtime

Untuk solusi hosting terkelola, buat server game Anda agar berjalan di salah satu gambar mesin Amazon berikut (AMIs). Lihat [Amazon GameLift ServersVersi AMI](#) Amazon GameLift Servers untuk lebih jelasnya.

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

Note

Amazon Linux 2 (AL2) akan mencapai akhir dukungan pada 30 Juni 2025. Lihat detail lebih lanjut di [Amazon Linux 2 FAQs](#). Untuk server game yang di-host AL2 dan menggunakan Amazon GameLift Servers server SDK 4.x, pertama-tama perbarui build server game ke server SDK 5.x, lalu terapkan ke 023 instance. AL2 Lihat [Migrasi ke server SDK 5.x untuk Amazon GameLift Servers](#).

Alat tambahan

[Pembungkus server game untuk Amazon GameLift Servers](#)

Alat ini membantu Anda menyebarkan server game untuk hosting dengan serangkaian fungsionalitas manajemen sesi permainan dasar. Dengan alat ini, Anda tidak perlu membuat perubahan pada kode game Anda atau mengintegrasikan SDK server untuk Amazon GameLift Servers. Gunakan pembungkus server game untuk mengemas server game Anda dan menyebarkannya untuk hosting game menggunakan salah satu dari tiga solusi Amazon GameLift Servers hosting (Di mana saja, terkelola EC2, atau kontainer terkelola). Alat ini paling cocok untuk evaluasi awal atau pembuatan prototipe dengan game Anda sendiri atau game contoh, karena alat ini tidak mendukung kustomisasi server game. Jika game Anda tidak memerlukan fitur khusus, Anda dapat menggunakan server game Anda dengan pembungkus server game untuk hosting produksi.

[Amazon GameLift Servers Toolkit](#)

Amazon GameLift Servers Toolkit adalah kumpulan skrip dan alat lain yang telah kami kembangkan untuk membantu pengembang dengan skenario dan masalah umum. Materi toolkit termasuk skrip, kode sampel, dan readmes.

- [Kit starter kontainer](#) - Gunakan alat ini untuk merampingkan tugas menyiapkan build server game untuk hosting dengan kontainer Amazon GameLift Servers terkelola. Kit ini mengintegrasikan fitur manajemen sesi permainan penting ke dalam server game, dan menggunakan templat yang telah dikonfigurasi sebelumnya untuk membuat armada kontainer dan menyiapkan pipeline penerapan

otomatis untuk pembuatan server game Anda. Setelah penerapan, Anda dapat memantau kinerja armada, mengelola sesi game, dan menganalisis metrik menggunakan Amazon GameLift Servers konsol dan alat API. Kit ini terintegrasi dengan AWS CodeBuild otomatisasi build, Amazon Simple Storage Service untuk penyimpanan, dan AWS CloudFormation untuk penyebaran infrastruktur.

- [Alat pembaruan build cepat](#) — Gunakan alat ini untuk memodifikasi build server game yang sudah diterapkan ke EC2 armada terkelola. Alat ini dibuat untuk membantu Anda dengan cepat menukar file build game tanpa harus mengkonfigurasi dan membuat EC2 armada baru dengan setiap perubahan. Anda dapat memperbarui instance individual atau semua instance di armada. Opsi memungkinkan Anda mengganti file build tertentu atau seluruh build, dan memungkinkan Anda mengelola cara memulai ulang server game setelah pembaruan.

Untuk layanan klien game

Buat layanan backend 64-bit untuk klien game Anda menggunakan AWS SDK, yang mencakup API layanan untuk. Amazon GameLift Servers Layanan backend game Anda menangani interaksi sisi klien dengan Amazon GameLift Servers layanan, termasuk memulai sesi permainan baru dan bergabung dengan pemain ke game.

[Dapatkan AWS SDK](#)

Untuk informasi selengkapnya tentang menggunakan AWS SDK dengan Amazon GameLift Servers, lihat sumber daya berikut:

- [Referensi Amazon GameLift Servers API](#)
- [Integrasi layanan klien](#)
- [Desain layanan backend klien](#)

Untuk manajemen Amazon GameLift Servers sumber daya

Gunakan alat berikut untuk membuat, memperbarui, dan memantau sumber daya hosting Amazon GameLift Servers terkelola Anda.

- [AWS Management Console](#) AWS Konsol adalah aplikasi berbasis web yang menyediakan akses terpusat ke semua konsol AWS layanan individu, termasuk. Amazon GameLift Servers Gunakan Konsol untuk membuat atau masuk ke AWS akun dan membuka Amazon GameLift Servers konsol untuk bekerja dengan sumber daya hosting game Anda. Anda dapat mengonfigurasi dan menyebarkan armada hosting dan sumber daya lainnya, melihat metrik penggunaan dan kinerja,

melacak sumber daya di dasbor, dan banyak tugas lainnya. [Pergi ke Amazon GameLift Servers konsol.](#)

- [API Layanan untuk Amazon GameLift Servers](#) — API ini memberi Anda akses terprogram ke semua Amazon GameLift Servers sumber daya Anda. Ini adalah bagian dari AWS SDK, yang dapat Anda unduh untuk digunakan dengan bahasa pemrograman paling populer. [Dapatkan AWS SDK.](#)
- [AWS antarmuka baris perintah \(CLI\)](#) - AWS CLI memungkinkan Anda berinteraksi dengan AWS layanan menggunakan shell baris perintah. Alat ini menyediakan akses langsung ke publik APIs untuk AWS layanan serta perintah khusus yang tersedia untuk layanan. [Dapatkan AWS CLI.](#)
- [AWS CloudFormation](#) for Amazon GameLift Servers — AWS CloudFormation Layanan ini membantu Anda memodelkan dan menyiapkan AWS sumber daya untuk merampingkan penyebaran dan manajemen infrastruktur. Buat AWS CloudFormation template untuk menjelaskan Amazon GameLift Servers sumber daya untuk solusi hosting Anda, lalu gunakan template untuk membangun sumber daya tambahan atau memperbarui konfigurasi. Lihat [referensi jenis Amazon GameLift Servers sumber daya.](#)

Untuk Amazon GameLift ServersRealtime

Konfigurasi dan terapkan Realtime server untuk meng-host game multipemain Anda. Untuk memungkinkan klien game Anda terhubung ke Realtime server, gunakan SDK Amazon GameLift Servers Realtime klien. Untuk memulai, [unduh SDK Realtime klien](#). Untuk informasi konfigurasi, lihat [Mengintegrasikan klien game untuk Amazon GameLift ServersRealtime](#).

Dukungan SDK

SDK Realtime Klien berisi sumber untuk bahasa berikut:

- C# (.NET)

Lingkungan pengembangan

Bangun SDK dari sumber sesuai kebutuhan untuk sistem operasi pengembangan dan mesin game berikut yang didukung:

- Sistem operasi - Windows, Linux, Android, iOS
- Mesin game – Unity, mesin yang mendukung pustaka C#

Sistem operasi server game

Anda dapat menyebarkan Realtime server ke sumber daya hosting yang berjalan di platform berikut:

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

Note

AL2 Mendekati akhir dukungan. Lihat detail lebih lanjut di [Amazon Linux 2 FAQs](#).

Tutorial: Orientasi cepat dengan pembungkusnya Amazon GameLift Servers

Selamat datang di tutorial orientasi untuk Amazon GameLift Servers. Dalam tutorial ini, Anda dengan cepat menyebarkan server game Anda untuk di-host pada armada sumber daya komputasi berbasis cloud. Gunakan tutorial ini untuk melewati pekerjaan mengintegrasikan SDK server Amazon GameLift Servers ke dalam kode game Anda, dan sebagai gantinya menyebarkan game Anda dengan fungsionalitas minimal untuk berkomunikasi dengan Amazon GameLift Servers layanan dan menjalankan sesi permainan. Anda akan menyiapkan solusi hosting dasar dan menggunakannya untuk mengalami berbagai fitur seperti penskalaan otomatis dan perijodohan. Ini juga merupakan cara yang bagus untuk meng-host prototipe game Anda sebagai bagian dari demo langsung atau untuk pengujian.

Manfaat utama dari metode orientasi ini:

- Gunakan server game Anda dengan cepat untuk hosting dengan cepat.
- Tidak ada perubahan kode game dan tidak ada modifikasi yang diperlukan.
- Gunakan metode ini dengan game apa pun yang dapat dieksekusi, terlepas dari mesin game.
- Jelajahi semua alat Amazon GameLift Servers manajemen, termasuk pemantauan aktivitas sesi game dan kesehatan hosting.

 Note

Pembungkus dimaksudkan untuk evaluasi dan penggunaan produksi dasar. Fitur-fitur canggih seperti manajemen sesi pemain terperinci memerlukan integrasi SDK server penuh.

Prasyarat

Sebelum Anda mulai, pastikan Anda memiliki:

- A Akun AWS dengan izin yang sesuai
- AWS CLI dipasang
- Pergi 1.18+
- Server game multipemain yang dapat dieksekusi
- Membuat (Linux/Mac)
- Git diinstal dengan akun aktif

Gambaran Umum

Dalam tutorial ini Anda akan:

1. Dapatkan dan bangun pembungkusnya
2. Siapkan game build
3. Konfigurasi pembungkusnya
4. Unggah build server game
5. Buat EC2 armada yang dikelola
6. Buat dan sambungkan ke sesi permainan
7. Pantau dan kelola server game Anda
8. Skala server game Anda

Langkah 1: Dapatkan dan bangun pembungkus server game

Gunakan perintah berikut untuk mendapatkan sumber pembungkus server game dan membangun pembungkusnya. Perintah ini menggunakan SSH, tetapi Anda juga dapat pergi ke repositori Github secara langsung.

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 dan Linux

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

Pada build yang berhasil, direktori “out” ditambahkan ke `amazon-gamelift-servers-game-server-wrapper`. Dalam direktori ini ada tiga folder, satu untuk setiap opsi armada hosting yang didukung, yang berisi satu set artefak build. Untuk tutorial ini, Anda menyebarkan ke EC2 armada terkelola, jadi Anda akan menggunakan folder tersebut. `gamelift-servers-managed-ec2`

Langkah 2: Siapkan build server game Anda

Pada langkah ini, Anda menyiapkan file build server game Anda untuk diunggah ke file. Amazon GameLift Servers

Buat direktori game

Sekarang, siapkan direktori game di mesin lokal Anda. Direktori ini harus berisi semua file yang Anda butuhkan untuk menjalankan server game Anda Amazon GameLift Servers. Ini termasuk pembungkus server game, build server game Anda, dan `config.yaml` yang membuat pembungkus berfungsi dengan server game Anda.

Gunakan langkah-langkah berikut:

1. EC2 Armada yang dikelola. Di folder keluaran pembungkus server game, temukan artefak build untuk digunakan ke armada Terkelola. EC2 Build akan ditulis ke direktori out seperti yang ditunjukkan di sini: `out\linux\amd64\gamelift-servers-managed-ec2`.

2. Salin server game Anda yang dapat dieksekusi dan semua file terkait yang diperlukan untuk masuk ke folder. `gamelift-servers-managed-ec2` Anda dapat memiliki direktori bersarang sesuai kebutuhan.

Contoh struktur direktori akan terlihat seperti ini:

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

Langkah 3: Konfigurasikan pembungkus untuk armada Anda

Amazon GameLift Servers mengelola siklus hidup instance komputasi armada, memutar instans baru dengan build server yang diinstal dan mendaur ulang instans sesuai kebutuhan. Layanan ini mengelola siklus hidup proses server game yang berjalan pada setiap instance. EC2 Armada yang Dikelola dapat memiliki instance di beberapa lokasi untuk mendukung pemain di mana pun mereka berada.

Edit file `config.yaml` untuk mengonfigurasi pembungkus untuk logging, pengaturan port, dan inisialisasi server.

1. Konfigurasikan pengaturan logging. Pembungkus server game menghasilkan pesan log untuk setiap proses server game. Secara default, level log diatur ke debug untuk verbositas maksimum. Ini sangat berguna selama pengaturan dan pemecahan masalah dan menentukan seberapa detail pesan log - dalam hal ini, yang paling bertele-tele. Opsi termasuk debug, info, peringatan, dan kesalahan (paling tidak bertele-tele).
2. Tentukan jalur ke direktori log server game. Jalur default untuk log server game adalah `./game-server-logs`. Direktori ini berisi semua log yang dihasilkan oleh server game Anda dan setiap instance memilikinya. Log secara otomatis diunggah ke Amazon GameLift Servers tempat mereka diakses dari tab Acara. Lihat bagian Pemecahan Masalah untuk detail selengkapnya.
3. Tentukan konfigurasi port jaringan. Atur port game ke apa pun yang Anda inginkan. Untuk tutorial ini, tentukan satu port saja, karena Anda akan membuat armada yang hanya menjalankan

satu proses server game bersamaan per instance. Jika Anda memutuskan untuk menjalankan beberapa proses sekaligus, Anda harus mengonfigurasi port yang cukup untuk setiap proses bersamaan. Nilai defaultnya adalah 37016 seperti yang ditunjukkan pada file konfigurasi, tetapi secara umum, untuk armada yang menggunakan build Linux menggunakan port 22 dan 1026-60000. Untuk armada yang menggunakan build Windows, gunakan port 1026-60000.

4. Siapkan jalur ke server game yang dapat dieksekusi. Untuk `./MyGame/my-server-executable.exe` menyesuaikan jalur ke server game Anda yang dapat dieksekusi dengan nama dan lokasi sebenarnya untuk itu. Ini adalah titik masuk untuk meluncurkan server game Anda.
5. Konfigurasi argumen server game. Minimal, tentukan argumen `-port` dan gunakan nilai port game yang sama yang Anda tentukan sebelumnya. Nilai "pos" 0 menunjukkan ini adalah argumen pertama. Tambahkan argumen lain sesuai kebutuhan. Argumen ini diteruskan ke server game Anda saat diluncurkan, memungkinkan Anda mengonfigurasi perilaku runtime-nya.
 - a. Argumen: `--port`
 - b. Nilai: `"{{.port number here}}"`
 - c. Posisi: 0 (Argumen pertama dalam daftar)

Contoh konfigurasi:

```
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
```

Langkah 4: Unggah build server game

Anda sekarang telah menyelesaikan semua elemen yang diperlukan dari pembuatan server game Anda (pembungkus server game, config.yaml, dan file server game Anda), dan Anda siap mengunggah build game Anda untuk hosting. Amazon GameLift Servers Cara tercepat untuk

mengunggah build game Anda adalah dengan menggunakan AWS CLI perintah `upload-build` seperti yang ditunjukkan pada contoh berikut.

Mengunggah build game dengan 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

Untuk build Mac dan Linux, gunakan `--operating-system AMAZON_LINUX_2023`

Saat Anda membuat build, rekam ID build dari respons API untuk digunakan untuk pembuatan armada.

Langkah 5: Buat EC2 armada yang dikelola

Langkah-langkah berikut menjelaskan konfigurasi armada minimal, sehingga Anda dapat aktif dan berjalan sesegera mungkin.

Untuk membuat armada Anda:

1. Masuk ke AWS Management Console dan navigasikan ke Amazon GameLift Servers.
2. Pada bilah menu di bagian atas jendela konsol, periksa untuk memastikan wilayah build Anda berada. Catat itu karena armada Anda harus berada di wilayah yang sama atau Anda tidak akan dapat menemukan atau memilih bangunan Anda.
3. Di panel navigasi di EC2 bagian Dikelola, pilih Builds.
4. Pilih build yang Anda unggah sebelumnya untuk menampilkan halaman Detail Build.
5. Di bagian Armada pilih Buat Armada yang menampilkan halaman Tentukan detail EC2 armada Terkelola tempat Anda dapat melacak status armada Anda serta melihat peristiwa pembuatan armada di tab Acara.

6. Isi nama dan deskripsi dan pilih Berikutnya.
7. Di halaman Tentukan detail instance, wilayah build ditampilkan secara default. Pilih wilayah tambahan yang ingin Anda tambahkan.
8. Untuk tipe Armada pilih On-Demand.
9. Di bawah Jenis instans pilih c5.large dan pilih Berikutnya.
- 10 Di bawah konfigurasi Runtime, Karena build game yang Anda unggah menggunakan wrapper, Anda harus menunjuk ke wrapper executable sebagai gantinya. Untuk server game Windows iniC : \game\amazon-gamelift-servers-game-server-wrapper.exe. Untuk server game Linux, ini/local/game/amazon-gamelift-servers-game-server-wrapper.

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

Selain itu, konfigurasikan nilai port game dengan rentang yang memungkinkan port yang ditetapkan dalam parameter peluncuran konfigurasi runtime config.yaml dan runtime. Port di config.yaml tidak perlu cocok dengan yang ditentukan dalam konfigurasi runtime armada, tetapi saat runtime jika konfigurasi menentukan port yang berbeda, nilai itu menggantikan apa yang ada di port. config.yaml Parameter peluncuran yang dimasukkan dalam konfigurasi runtime juga mengganti apa yang ada di config.yaml

- 11 Pada halaman Tinjau dan buat, periksa ulang semua konfigurasi, lalu pilih Kirim untuk membuat armada Anda. Status armada akan berubah karena meningkatkan kapasitas untuk meng-host server game, dan segera menunjukkan status Aktif. Setelah aktivasi selesai dan armada dikerahkan, layanan meluncurkan pembungkus yang siap menerima permintaan sesi permainan.

Langkah 6: Buat dan sambungkan ke sesi permainan

Ketika status armada Anda menunjukkan Aktif, ini berarti server game siap dan menunggu untuk menyelenggarakan sesi permainan. Untuk memulai sesi permainan, buat permintaan sesi permainan ke Amazon GameLift Servers layanan. Di sini, Anda akan menggunakan AWS CLI untuk membuat permintaan ini.

Note

Perlu diingat bahwa membuat sesi permainan menggunakan AWS CLI berguna untuk menguji dan menjadi akrab dengan prosesnya. Pada titik tertentu, Anda akan menambahkan

panggilan AWS SDK terprogram ke layanan backend game Anda sebagai bagian dari sistem perijodohan atau penempatan sesi game Anda.

Gunakan yang berikut ini untuk membuat sesi permainan:

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

Anda juga dapat meneruskan properti game yang disesuaikan ke server yang dapat dieksekusi. Lihat Argumen Server Game di README untuk detailnya. Ketika menerima create-game-session panggilan, Amazon GameLift Servers menginformasikan pembungkus untuk meluncurkan server game yang dapat dieksekusi dan memulai sesi permainan. Apa yang ada di `config.yaml` mempengaruhi konfigurasi server game, dan parameter peluncuran yang diatur di konsol menentukan konfigurasi sesi game itu sendiri.

Contoh format untuk menambahkan properti game:

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

Langkah 7: Kelola dan pantau armada Anda

Sekarang armada server game Anda sudah diatur dan sesi permainan dimulai, Anda dapat mengelola dan memantaunya dari Amazon GameLift Servers konsol. Cara terbaik untuk

melakukannya adalah dari halaman detail Armada tempat Anda dapat mengedit detail armada atau mengubah penskalaan dan kapasitas armada di tab Penskalaan. Lihat bagian berikut tentang penskalaan server game Anda.

Pilih tab Metrik untuk melihat grafik yang menggambarkan pemantauan untuk metrik Aktivitas dan Perangkat Keras. Untuk detail tentang grafik metrik, pilih tautan Info di sebelah ID armada Anda. Selain itu, meskipun Anda dapat memonitor server game Anda dari tab Metrik, Anda juga dapat menambahkan alarm ke metrik ini di dasbor. CloudWatch

Untuk menavigasi ke CloudWatch dasbor dari konsol:

1. Ketik CloudWatch "" di bilah pencarian dan pilih dari daftar hasil pencarian untuk menampilkan CloudWatch Ikhtisar.
2. Gulir ke bawah dan pilih Lihat GameLift dasbor untuk melihat grafik berdasarkan metrik utama untuk armada dan sesi permainan Anda.

Langkah 8: Skala server game Anda

Untuk langkah selanjutnya, Anda mengatur penskalaan otomatis. Dengan penskalaan otomatis, Anda secara dinamis menskalakan kapasitas armada Anda sebagai respons terhadap aktivitas server game. Saat pemain tiba dan memulai sesi permainan, penskalaan otomatis menambahkan lebih banyak instance, dan saat permintaan pemain menurun, penskalaan otomatis melepaskan instance yang tidak digunakan. Ini adalah cara yang efektif untuk meminimalkan sumber daya dan biaya hosting Anda sambil tetap memberikan pengalaman pemain yang lancar dan cepat.

Saat Anda mempersiapkan peluncuran game, Anda akan ingin mengatur penskalaan otomatis untuk armada Anda. Penskalaan otomatis direkomendasikan sebagai cara yang efektif untuk meminimalkan sumber daya dan biaya hosting Anda sambil tetap memberikan pengalaman pemain yang lancar dan cepat.

Untuk mengatur kapasitas armada secara manual

Untuk mengatur kapasitas armada secara manual

1. Buka tab Penskalaan di halaman detail untuk armada Anda.
2. Pilih lokasi dan pilih Edit.
3. Ubah nilai instans yang diinginkan dan ubah setelan ukuran Min dan Maks untuk skala melampaui nilai saat ini.

 Note

Gunakan pengaturan instans maks sebagai stop gap untuk mencegah penskalaan dan biaya yang berlebihan.

4. Pilih Konfirmasi.

Untuk menggunakan penskalaan otomatis berbasis target

Untuk menggunakan penskalaan otomatis berbasis target

Penskalaan otomatis berbasis target (pelacakan target) menghubungkan penskalaan armada ke persentase sesi permainan yang tersedia. Saat pemain melonjak untuk memainkan game Anda dan sesi permainan yang tersedia berkurang, itu merespons dengan secara otomatis menambahkan lebih banyak instance ke armada.

1. Di bawah kebijakan penskalaan otomatis berbasis target, pilih Tambahkan kebijakan dan atur kapasitas armada untuk berubah secara otomatis saat mencapai ambang persentase sesi game yang tersedia yang Anda tetapkan. Buffer yang lebih besar dapat menangani lonjakan dengan lebih baik, membuat pemain baru masuk ke game dengan cepat, tetapi juga dapat menyebabkan biaya hosting yang lebih tinggi.
2. Pilih Konfirmasi untuk menerima perubahan.

Penskalaan otomatis berbasis aturan memberi Anda kontrol yang lebih terperinci, seperti kemampuan untuk menautkan penskalaan ke metrik armada lainnya dan menetapkan ambang batas khusus dan respons penskalaan. Ini menawarkan opsi yang kuat, tetapi juga membutuhkan penggunaan CLI dan pengujian signifikan untuk memahami bagaimana aturan khusus berperilaku dalam tindakan. Tutorial ini berfokus pada pengaturan pertama pendekatan berbasis target.

Memecahkan masalah umum

Berikut ini adalah masalah umum yang mungkin Anda temui saat menjalankan server game dan sesi game Anda. Ketika server atau sesi game tidak berjalan dengan baik, langkah pertama adalah memeriksa log yang mungkin mengungkapkan salah satu masalah yang dijelaskan di bawah ini untuk penerapan atau game baru dalam produksi.

Berikut ini sering terungkap dalam log:

- Proses server game tidak dapat diluncurkan. Ini mungkin kesalahan dalam konfigurasi pembungkus—verifikasi file memiliki jalur peluncuran yang benar dan parameter serta argumen peluncuran yang benar.
- Pembuatan server game tidak dapat dijalankan. Ini kemungkinan kesalahan dalam kode game.
- Pemain tidak dapat terhubung ke sesi permainan. Ini kemungkinan kesalahan konfigurasi port.
- Lag atau koneksi lambat. Tinjau kebijakan dan ambang batas penskalaan.
- Tidak ada koneksi. Verifikasi aturan dan konfigurasi port untuk armada Anda.

Untuk melihat log peristiwa untuk Amazon GameLift Servers armada Anda

Untuk melihat log peristiwa untuk Amazon GameLift Servers armada Anda

1. Buka konsol Amazon GameLift Servers.
2. Pada halaman Detail Armada pilih tab Acara dan unduh log. Anda juga dapat memantau metrik aktivitas dan perangkat keras untuk kesehatan server game dan aktivasi sesi game dari tab Metrik.

Untuk melihat log sesi permainan

Untuk melihat log sesi permainan

1. Dari konsol, buka armada Anda dan buka tab Sesi permainan.
2. Pilih ID sesi game dari daftar untuk menampilkan halaman Ikhtisar untuknya.
3. Pilih Unduh log untuk mengunduh file log secara lokal.

Untuk melihat log sesi game dengan CLI gunakan API. `GetGameSessionLogURL` Amazon GameLift Servers secara otomatis menyimpan log selama 14 hari.

Anda juga dapat mengonfigurasi CloudWatch log Amazon untuk armada Anda. Ini memberikan kemampuan logging tambahan dan integrasi dengan layanan AWS pemantauan lainnya.

Untuk akses log waktu nyata atau periode retensi yang diperpanjang melalui CloudWatch:

1. Di bagian atas dasbor Amazon GameLift Servers konsol Anda ketik CloudWatch "" di bilah pencarian dan pilih dari drop-down hasil.
2. Buka Grup CloudWatch Log dan cari sesi tertentu. Metode termudah adalah dengan mengklik Cari Semua dan filter menggunakan `gameSessionId` atau `ClientId`.

Langkah selanjutnya

- [Buat armada multi-lokasi untuk menambahkan hosting di lokasi tambahan](#)
- [Tambahkan antrean sesi permainan untuk membuat penempatan terbaik untuk sesi permainan di beberapa lokasi](#)
- [Bereksperimenlah dengan FlexMatch perjodohan dengan membuat mak comblang dan aturan yang ditetapkan untuk gim Anda](#)
- [Mulai kerjakan fungsionalitas untuk klien game dan komponen layanan backend Anda, sehingga pemain dapat membuat permintaan bergabung dan langsung terhubung ke sesi game](#)
- [Saat siap, pindah ke solusi yang terintegrasi penuh](#)

Jelajahi dengan Amazon GameLift Servers plugin

Bagian Amazon GameLift Servers plugin adalah add-on berfitur lengkap untuk mesin game Unreal atau Unity Anda. Ini memandu Anda melalui langkah-langkah dasar untuk menyebarkan game Anda untuk hosting Amazon GameLift Servers. Dengan set alat dan alur kerja plugin, Anda dapat bekerja di lingkungan pengembangan mesin game Anda untuk mempersiapkan server game Anda untuk hosting, mengatur hosting di mesin lokal untuk pengujian, membuat layanan backend sederhana, dan menyebarkan server game Anda ke hosting berbasis cloud yang dikelola.

Gunakan plugin untuk pengalaman bekerja dengan Amazon GameLift Servers dan dapatkan solusi hosting game dan berjalan cepat. Anda dapat bekerja dengan aset game sampel atau proyek game Anda sendiri. Plugin ini mengotomatiskan sejumlah langkah sehingga Anda dapat dengan cepat membangun solusi kerja yang sederhana. Saat Anda menyelesaikan alur kerja terpandu plugin, Anda akan dapat menghubungkan klien game ke sesi game yang dihosting langsung melalui Amazon GameLift Servers. Setelah menggunakan plugin untuk membuat solusi hosting sederhana, Anda dapat menyesuaikan solusi Anda untuk memenuhi kebutuhan game Anda.

Plugin ini tersedia untuk mesin game berikut:

- Unreal Engine
- Unity

Plugin ini mencakup komponen-komponen ini untuk setiap mesin game:

- Modul plugin untuk editor mesin game. Ketika plugin diinstal, tombol menu utama baru memberi Anda akses ke Amazon GameLift Servers fungsionalitas.
- Perpustakaan untuk Amazon GameLift Servers API layanan dengan fungsionalitas sisi klien.
- Perpustakaan untuk Amazon GameLift Servers server SDK (versi 5).
- Contoh aset untuk digunakan dengan menguji integrasi server.
- Konfigurasi yang dapat diedit, dalam bentuk AWS CloudFormation templat, yang menentukan solusi server game Anda.

Topik

- [Alur kerja plugin](#)
- [Amazon GameLift Serversplugin untuk Unreal Engine](#)
- [Amazon GameLift Serversplugin untuk Unity \(server SDK 5.x\)](#)
- [Amazon GameLift Serversplugin untuk Unity \(server SDK 4.x\)](#)

Alur kerja plugin

Langkah-langkah berikut menjelaskan jalur khas untuk mempersiapkan dan menyebarkan proyek game Anda Amazon GameLift Servers. Anda menyelesaikan langkah-langkah ini dengan bekerja di editor mesin game dan kode game Anda.

1. Buat profil pengguna yang menautkan ke pengguna AWS akun Anda dan menyediakan kredensi akses dengan izin untuk digunakan Amazon GameLift Servers.
2. Siapkan AWS sumber daya terkait yang digunakan plugin dalam solusi hosting (disebut sebagai “bootstrap”).
3. Tambahkan kode server ke proyek Anda untuk membangun komunikasi antara server game yang sedang berjalan dan Amazon GameLift Servers layanan.
4. Tambahkan kode klien ke proyek Anda yang memungkinkan klien game mengirim permintaan Amazon GameLift Servers untuk memulai sesi permainan baru dan kemudian terhubung ke mereka.
5. Gunakan alur kerja Anywhere untuk menyiapkan workstation lokal Anda sebagai komputasi Anywhere dan host server game Anda. Luncurkan server game dan klien Anda secara lokal melalui plugin, sambungkan ke sesi game, dan uji integrasi.
6. Gunakan EC2 alur kerja Terkelola untuk mengunggah server game Anda Amazon GameLift Servers dan menerapkan solusi cloud hosting yang sederhana namun lengkap. Luncurkan klien

game Anda secara lokal melalui plugin, minta sesi game dan sambungkan ke sana, dan mainkan game Anda.

Saat bekerja di plugin, Anda akan membuat dan menggunakan AWS sumber daya, Tindakan ini mungkin dikenakan biaya ke AWS akun yang digunakan. Jika Anda baru mengenal AWS, tindakan ini mungkin tercakup dalam [Tingkat AWS Gratis](#).

Amazon GameLift Serversplugin untuk Unreal Engine

Plugin ini menambahkan SDK server Amazon GameLift Servers C ++ dan alat ke editor UE. Gunakan alur kerja UI terpandu untuk mengintegrasikan fungsionalitas SDK server ke dalam proyek game Anda dan menerapkan solusi Amazon GameLift Servers hosting untuk server game Anda.

Dengan plugin, Anda dapat membangun solusi hosting kerja dasar dan kemudian mengoptimalkan dan menyesuaikan sesuai kebutuhan. Siapkan armada Amazon GameLift Servers Anywhere dengan workstation lokal Anda sebagai host. Untuk hosting cloud dengan armada kontainer terkelola EC2 atau terkelola, gunakan server game Anda dengan solusi lengkap untuk mengelola permintaan sesi game dan koneksi klien.

Topik

- [Instal plugin untuk proyek game Unreal Anda](#)
- [Langkah selanjutnya: Sesuaikan solusi hosting game Anda](#)
- [Plugin untuk Unreal: Siapkan profil AWS pengguna](#)
- [Plugin untuk Unreal: Integrasikan kode game Anda](#)
- [Plugin untuk Unreal: Tuan rumah game Anda secara lokal dengan Anywhere Amazon GameLift Servers](#)
- [Plugin untuk Unreal: Menyebar game Anda ke armada yang dikelola EC2](#)
- [Plugin untuk Unreal: Menyebar game Anda ke armada kontainer terkelola](#)

Instal plugin untuk proyek game Unreal Anda

[Dapatkan Amazon GameLift Servers plugin untuk Unreal Engine dari GitHub](#)

Lihat readme GitHub repositori untuk informasi tentang cara menginstal plugin di Unreal Editor Anda untuk proyek game.

Plugin ini mencakup komponen-komponen ini:

- Modul plugin untuk editor UE. Ketika plugin diinstal, tombol menu utama baru memberi Anda akses ke Amazon GameLift Servers fungsionalitas.
- Pustaka C++ untuk API Amazon GameLift Servers layanan. Gunakan fungsionalitas API dalam layanan backend sisi klien untuk membantu klien game meminta sesi game dan send/retrieve informasi sesi game.
- Pustaka tidak nyata untuk SDK Amazon GameLift Servers server (versi 5). Gunakan SDK server dalam kode server game Anda untuk mengelola komunikasi antara proses server game yang dihosting dan Amazon GameLift Servers layanan.
- Konten untuk pengujian, termasuk peta game startup dan dua peta pengujian dengan cetak biru dasar dan elemen UI untuk digunakan dengan menguji integrasi server.
- Konfigurasi yang dapat diedit, dalam bentuk AWS CloudFormation templat, yang digunakan plugin saat menyebarkan server game Anda untuk hosting.

Plugin ini menggunakan AWS CloudFormation template untuk menyebarkan solusi hosting untuk skenario game umum. Gunakan solusi ini seperti yang disediakan atau sesuaikan sesuai kebutuhan untuk game Anda.

Langkah selanjutnya: Sesuaikan solusi hosting game Anda

Menggunakan alur kerja terpandu plugin adalah cara yang baik untuk bangun dan berjalan cepat dengan solusi Amazon GameLift Servers hosting. Dengan plugin, Anda dapat mengatur versi dasar dari setiap komponen solusi Anda.

Ketika Anda siap, Anda dapat membangun solusi dasar ini dengan menyesuaikan setiap komponen, dan menyempurnakan solusi Anda saat Anda mempersiapkan peluncuran game. Pertimbangkan opsi ini:

- Ubah konfigurasi armada dan armada Anda. Lihat [Sesuaikan Amazon GameLift Servers EC2 armada yang dikelola](#).
- Sesuaikan konfigurasi antrian sesi permainan Anda. Lihat [Kustomisasi antrian sesi permainan](#).
- Tambahkan fungsionalitas ke server game dan klien game Anda. Lihat [Integrasikan server game Anda dengan Amazon GameLift Servers](#) dan [Integrasikan klien game Anda dengan Amazon GameLift Servers](#).
- Sesuaikan layanan backend Anda. Lihat [Rancang layanan klien game Anda](#).

- Siapkan penskalaan kapasitas otomatis untuk memenuhi permintaan pemain yang diharapkan. Lihat [Menskalakan kapasitas hosting game dengan Amazon GameLift Servers](#).
- Siapkan alat observabilitas hosting, termasuk analitik dan pencatatan. Lihat [Pemantauan Amazon GameLift Servers](#).
- Otomatiskan penerapan Anda menggunakan [infrastruktur sebagai kode \(IaC\)](#). Alur kerja terpandu plugin untuk solusi terkelola menggunakan AWS CloudFormation templat. Anda dapat menyesuaikan ini sesuai kebutuhan. Lihat [Mengelola Amazon GameLift Servers sumber daya hosting menggunakan AWS CloudFormation](#).

Topik

- [Plugin untuk Unreal: Siapkan profil AWS pengguna](#)
- [Plugin untuk Unreal: Integrasikan kode game Anda](#)
- [Plugin untuk Unreal: Tuan rumah game Anda secara lokal dengan Anywhere Amazon GameLift Servers](#)
- [Plugin untuk Unreal: Menyebar game Anda ke armada yang dikelola EC2](#)
- [Plugin untuk Unreal: Menyebar game Anda ke armada kontainer terkelola](#)

Plugin untuk Unreal: Siapkan profil AWS pengguna

Setelah menginstal plugin, atur profil pengguna dengan AWS akun yang valid. Anda dapat mempertahankan beberapa profil di plugin, tetapi Anda hanya dapat memiliki satu profil yang dipilih pada satu waktu. Setiap kali Anda bekerja di plugin, pilih profil yang akan digunakan. Setiap halaman alur kerja menampilkan profil yang dipilih saat ini.

Mempertahankan beberapa profil memberi Anda kemampuan untuk beralih di antara penerapan hosting yang berbeda. Misalnya, Anda dapat mengatur profil yang menggunakan AWS akun yang sama tetapi menyebar ke AWS Wilayah yang berbeda. Atau Anda dapat mengatur profil dengan AWS akun atau pengguna yang berbeda dan set izin.

Note

Jika Anda telah menginstal AWS CLI di workstation Anda dan memiliki profil yang sudah dikonfigurasi, Amazon GameLift Servers plugin akan mendeteksi dan mencantumkannya sebagai profil yang ada. Plugin secara otomatis memilih profil apa pun yang diberi nama[default]. Anda dapat menggunakan profil yang ada atau membuat yang baru.

Semua profil harus di-bootstrap untuk menyiapkan beberapa AWS sumber daya yang diperlukan di bawah pengguna akun Anda.

Untuk mengelola AWS profil Anda

1. Di toolbar utama editor Unreal, pilih Amazon GameLift Servers menu, dan pilih AWS Access Credentials. Tindakan ini membuka Amazon GameLift Servers plugin ke halaman Siapkan profil pengguna Anda.
2. Gunakan tombol untuk membuat AWS akun baru atau mengatur profil pengguna untuk AWS akun yang sudah Anda miliki.
3. Jika Anda belum memiliki profil pengguna, Anda diminta untuk memasukkan detail profil dan membuat profil baru. Saat diminta, berikan informasi berikut:
 - Sebuah AWS akun. Jika Anda membuat AWS akun baru, gunakan tautan ke AWS Management Console dan ikuti petunjuknya. Lihat [Membuat AWS akun](#) untuk detail selengkapnya.
 - AWS Pengguna dengan izin untuk menggunakan Amazon GameLift Servers dan AWS layanan lain yang diperlukan. Lihat [Mengatur sebuah Akun AWS](#) petunjuk tentang cara menyiapkan pengguna AWS Identity and Access Management (IAM) dengan Amazon GameLift Servers izin dan akses terprogram dengan kredensial jangka panjang.
 - Kredensial untuk pengguna Anda AWS . Kredensial ini terdiri dari ID kunci AWS akses dan kunci AWS rahasia. Lihat [Dapatkan kunci akses Anda](#) untuk detail selengkapnya.
 - AWS wilayah. Ini adalah lokasi geografis tempat Anda ingin membuat AWS sumber daya untuk hosting. Selama pengembangan, kami sarankan menggunakan wilayah yang dekat dengan lokasi fisik Anda. Pilih wilayah dari daftar [AWS wilayah yang didukung](#).
4. Jika plugin mendeteksi profil yang ada, itu akan menampilkan daftar profil yang tersedia. Pilih profil yang ada dari daftar, atau pilih Tambahkan profil lain untuk membuat profil baru.

Bootstrap profil pengguna

Semua profil harus di-bootstrap untuk digunakan dengan plugin. Amazon GameLift Servers Bootstrapping membuat bucket Amazon S3 khusus untuk profil. Ini digunakan untuk menyimpan konfigurasi proyek, membangun artefak, dan dependensi lainnya. Bucket tidak dibagikan di antara profil lain.

Bootstrapping melibatkan pembuatan AWS sumber daya baru dan mungkin menimbulkan biaya.

Untuk bootstrap profil Anda:

1. Pada halaman AWS Access Credentials, periksa status bootstrap dari profil pengguna yang ingin Anda gunakan. Jika status bootstrap profil adalah “Tidak Aktif” dan tidak ada bucket S3 yang terdaftar, Anda perlu mem-bootstrap profil.
2. Pilih profil yang ingin Anda gunakan dan pilih profil Bootstrap.
3. Tunggu status bootstrap berubah menjadi “Aktif”. Hal ini dapat menghabiskan waktu beberapa menit.

Plugin untuk Unreal: Integrasikan kode game Anda

Sebelum Anda dapat menyebarkan server game Anda ke armada, Anda perlu membuat serangkaian pembaruan untuk kode game dan komponen game paket untuk digunakan dengan Amazon GameLift Servers layanan ini.

Topik ini berjalan melalui langkah-langkah untuk melakukan integrasi minimal. Untuk integrasi server, gunakan contoh kode yang disediakan untuk memperbarui mode permainan proyek Anda.

- [Siapkan target build dan aturan modul](#)
- [Perbarui kode server game Anda](#)
- [Integrasikan peta permainan klien Anda](#)
- [Package komponen game Anda](#)

Siapkan target build dan aturan modul

Ubah file proyek game Anda untuk menghasilkan komponen build dengan benar untuk digunakan Amazon GameLift Servers.

Untuk menambahkan target build klien dan server:

1. Buka file kode proyek game Anda dan cari file `.../Games/[your application name]Source/[your application name]Target.cs` file. Contoh: `.../Source/GameLiftUnrealAppTarget.cs`. (Jika Anda menggunakan Visual Studio, buka `.sln` file proyek.)
2. Salin file ini untuk membuat dua file target baru di `Source/` direktori.

- Target klien - Ubah nama file baru menjadi *[your application name]*Client.Target.cs. Edit konten untuk memperbarui nama kelas dan nilai tipe target, seperti yang diilustrasikan dalam kode contoh berikut:

```
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" );
    }
}
```

- Target server - Ubah nama file baru menjadi *[your application name]*Server.Target.cs. Edit konten untuk memperbarui nama kelas dan nilai tipe target, seperti yang diilustrasikan dalam kode contoh berikut:

```
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. Regenerasi file proyek Anda. Jika Anda menggunakan Visual Studio, Anda dapat mengklik kanan .uproject file proyek game Anda dan memilih Generate Visual Studio Project Files.

Untuk memperbarui aturan modul proyek game:

Perbarui aturan modul proyek game untuk mengambil ketergantungan pada plugin.

1. Buka file kode proyek game Anda dan cari file `.../Games/[your application name]Source/[your application name].Build.cs` file. Contoh: `.../Source/GameLiftUnrealApp.Build.cs`. (Jika Anda menggunakan Visual Studio, buka `.sln` file proyek.)
2. Temukan `ModuleRules` kelas dan perbarui seperti yang diilustrasikan dalam kode contoh berikut:

```
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. Setelah membuat file target baru dan memodifikasi aturan modul, bangun kembali proyek game Anda.

Perbarui kode server game Anda

Perbarui kode server game Anda untuk mengaktifkan komunikasi antara proses server game dan Amazon GameLift Servers layanan. Server game Anda harus dapat menanggapi permintaan dari Amazon GameLift Servers, seperti untuk memulai dan menghentikan sesi permainan baru.

Untuk menambahkan kode server untuk Amazon GameLift Servers

1. Di editor kode Anda, buka file solution (`.sln`) untuk proyek game Anda, biasanya ditemukan di folder root proyek. Sebagai contoh: `GameLiftUnrealApp.sln`.

2. Dengan solusi terbuka, cari file header mode game proyek: [project-name]GameMode.h file. Sebagai contoh: GameLiftUnrealAppGameMode.h.
3. Ubah file header agar sejajar dengan kode berikut. Pastikan untuk mengganti "GameLiftServer" dengan nama proyek Anda sendiri. Pembaruan ini khusus untuk server game; kami menyarankan Anda membuat salinan cadangan dari file mode permainan asli untuk digunakan dengan klien Anda.

Contoh kode 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;
};
```

- Buka file [project-name]GameMode.cpp file sumber terkait (misalnyaGameLiftUnrealAppGameMode.cpp). Ubah kode untuk menyelaraskan dengan kode contoh berikut. Pastikan untuk mengganti "GameLiftUnrealApp" dengan nama proyek Anda sendiri. Pembaruan ini khusus untuk server game; kami menyarankan Anda membuat salinan cadangan dari file asli untuk digunakan dengan klien Anda.

Kode contoh berikut menunjukkan cara menambahkan elemen minimum yang diperlukan untuk integrasi server denganAmazon GameLift Servers:

- Inisialisasi klien Amazon GameLift Servers API. InitSDK() Panggilan dengan parameter server diperlukan untuk armada Amazon GameLift Servers Anywhere. Saat Anda terhubung ke armada Anywhere, plugin menyimpan parameter server sebagai argumen konsol Kode sampel dapat mengakses nilai saat runtime.
- Menerapkan fungsi callback yang diperlukan untuk menanggapi permintaan dari Amazon GameLift Servers layanan, termasukOnStartGameSession,OnProcessTerminate, danonHealthCheck.
- Panggil ProcessReady() dengan port yang ditunjuk untuk memberi tahu Amazon GameLift Servers layanan saat siap menyelenggarakan sesi permainan.

Contoh kode server game

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

#include "GameLiftUnrealAppGameMode.h"

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

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

#include "GenericPlatform/GenericPlatformOutputDevices.h"

DEFINE_LOG_CATEGORY(GameServerLog);

AGameLiftUnrealAppGameMode::AGameLiftUnrealAppGameMode() :
    ProcessParameters(nullptr)
```

```

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

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

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

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

#ifdef WITH_GAMELIFT
    InitGameLift();
#endif
}

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

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

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

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

    if (bIsAnywhereActive)
    {

```



```

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

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

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

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

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

    FString glAnywhereAuthToken = "";

```

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

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

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

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

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

        UE_LOG(GameServerLog, SetColor, TEXT("%s"), COLOR_YELLOW);
        UE_LOG(GameServerLog, Log, TEXT(">>>> WebSocket URL: %s"),
*ServerParametersForAnywhere.m_webSocketUrl);
        UE_LOG(GameServerLog, Log, TEXT(">>>> Fleet ID: %s"),
*ServerParametersForAnywhere.m_fleetId);
```

```

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

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

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

    ProcessParameters = MakeShared<FProcessParameters>();

    //When a game session is created, Amazon GameLift Servers sends an activation
request to the game server and passes along the game session object containing game
properties and other settings.

```

```

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

//OnProcessTerminate callback. Amazon GameLift Servers will invoke this callback
before shutting down an instance hosting this game server.
//It gives this game server a chance to save its state, communicate with services,
etc., before being shut down.
//In this case, we simply tell Amazon GameLift Servers we are indeed going to
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  
}
```

Integrasikan peta permainan klien Anda

Peta game startup berisi logika cetak biru dan elemen UI yang sudah menyertakan kode dasar untuk meminta sesi game dan menggunakan informasi koneksi untuk terhubung ke sesi game. Anda dapat menggunakan peta apa adanya atau memodifikasinya sesuai kebutuhan. Gunakan peta game startup dengan aset game lainnya, seperti proyek template Orang Ketiga yang disediakan oleh Unreal Engine. Aset ini tersedia di Browser Konten. Anda dapat menggunakannya untuk menguji alur kerja penerapan plugin, atau sebagai panduan untuk membuat layanan backend khusus untuk game Anda.

Peta startup memiliki karakteristik sebagai berikut:

- Ini mencakup logika untuk armada Anywhere dan EC2 armada yang dikelola. Ketika Anda menjalankan klien Anda, Anda dapat memilih untuk terhubung ke salah satu armada.
- Fungsionalitas klien termasuk menemukan sesi permainan (`SearchGameSessions()`), membuat sesi permainan baru (`CreateGameSession()`), dan bergabung dengan sesi permainan secara langsung.
- Ini mendapat ID pemain unik dari kumpulan pengguna Amazon Cognito proyek Anda (ini adalah bagian dari solusi Anywhere yang diterapkan).

Untuk menggunakan peta game startup

1. Di editor UE, buka halaman Pengaturan Proyek, Peta & Mode, dan perluas bagian Peta Default.
2. Untuk Editor Startup Map, pilih "StartupMap" dari daftar dropdown. Anda mungkin perlu mencari file, yang terletak di... > Unreal Projects/[project-name]/Plugins/Amazon GameLift Servers Plugin Content/Maps.
3. Untuk Peta Default Game, pilih "StartupMap" yang sama dari daftar dropdown.
4. Untuk Peta Default Server, pilih "ThirdPersonMap". Ini adalah peta default yang disertakan dalam proyek game Anda. Peta ini dirancang untuk dua pemain dalam game.
5. Buka panel detail untuk peta default server. Setel GameMode Override ke "None".
6. Perluas bagian Mode Default, dan atur Mode Game Server Default Global ke mode permainan yang Anda perbarui untuk integrasi server Anda.

Setelah Anda membuat perubahan ini pada proyek Anda, Anda siap untuk membangun komponen game Anda.

Package komponen game Anda

Untuk mengemas server game dan build klien game Anda

1. Buka proyek game Anda dalam versi editor Unreal Engine yang dibuat sumber.
2. Gunakan editor untuk mengemas klien game dan server build Anda.
 - a. Pilih target. Buka Platform, Windows dan pilih salah satu dari yang berikut ini:
 - Peladen: `[your-application-name]Server`
 - Klien: `[your-application-name]Client`
 - b. Mulai membangun. Buka Platform, Windows, Package Project.

Setiap proses pengemasan menghasilkan executable: `[your-application-name]Client.exe` atau `[your-application-name]Server.exe`

Di plugin, atur jalur ke klien dan server build executable di workstation lokal Anda.

Plugin untuk Unreal: Tuan rumah game Anda secara lokal dengan Anywhere Amazon GameLift Servers

Gunakan alur kerja ini untuk menyiapkan workstation lokal Anda sebagai host server game menggunakan armada Anywhere. Anda dapat menggunakannya untuk menguji integrasi server game sebelum menerapkan ke armada terkelola berbasis cloud. Ini juga dapat berguna untuk pengujian lokal selama pengembangan game berulang.

Untuk memulai alur kerja Amazon GameLift Servers Anywhere:

- Di toolbar utama editor Unreal, pilih Amazon GameLift Servers menu, dan pilih Host with Anywhere. Tindakan ini membuka halaman plugin Deploy Anywhere, yang menyajikan proses enam langkah untuk mengintegrasikan, membangun, dan meluncurkan komponen game Anda.

Langkah 1: Atur profil Anda

Pilih profil yang ingin Anda gunakan saat mengikuti alur kerja ini. Profil yang Anda pilih memengaruhi semua langkah dalam alur kerja. Semua sumber daya yang Anda buat dikaitkan dengan AWS akun

profil dan ditempatkan di AWS Wilayah default profil. Izin pengguna profil menentukan akses Anda ke AWS sumber daya dan tindakan.

Untuk mengatur profil pengguna

1. Pilih profil dari daftar dropdown profil yang tersedia. Jika Anda belum memiliki profil atau ingin membuat yang baru, buka GameLift menu Amazon dan pilih Set AWS User Profiles.
2. Jika status bootstrap tidak “Aktif”, pilih profil Bootstrap dan tunggu statusnya berubah menjadi “Aktif”.

Langkah 2: Siapkan kode permainan Anda

Pada langkah ini, siapkan server game dan build klien game Anda untuk Amazon GameLift Servers dikerjakan. Jika Anda belum mengintegrasikan kode game Anda, lihat [Plugin untuk Unreal: Integrasikan kode game Anda](#). Masukkan jalur ke executable game Anda di workstation lokal Anda.

- Server game: Integrasikan server game Anda dengan SDK server untuk Amazon GameLift Servers dan paket build server game Anda. Untuk petunjuk, lihat [Plugin untuk Unreal: Integrasikan kode game Anda](#). Server game harus terintegrasi dengan SDK server untuk menjalin komunikasi dengan Amazon GameLift Servers layanan dan menanggapi permintaan untuk memulai sesi permainan baru dan menerima koneksi klien game.
- Klien game: Minimal, Anda memerlukan klien game yang dapat terhubung ke server game Anda menggunakan alamat IP dan informasi port. Jika komponen klien game belum disiapkan Amazon GameLift Servers, Anda dapat menggunakan AWS CLI alat ini untuk meminta sesi game baru secara manual, mendapatkan informasi koneksi, dan menggunakan informasi tersebut untuk menghubungkan klien game.

Pada titik tertentu, Anda harus memiliki layanan backend untuk mengirim permintaan sesi permainan baru ke Amazon GameLift Servers layanan dan menyampaikan informasi koneksi kembali ke klien game. Anda dapat menggunakan peta pengujian yang disertakan dengan plugin untuk menambahkan Amazon GameLift Servers fungsionalitas klien ke proyek game Anda. Untuk bantuan dalam membangun solusi khusus, lihat [Tambahkan Amazon GameLift Servers ke klien game Anda](#).

Langkah 3: Connect ke armada Anywhere

Pada langkah ini, Anda menunjuk armada Anywhere untuk digunakan. Armada Anywhere mendefinisikan kumpulan sumber daya komputasi, yang dapat ditemukan di mana saja, untuk hosting server game.

- Jika AWS akun yang saat ini Anda gunakan memiliki armada Anywhere yang sudah ada, buka bidang tarik-turun nama Armada dan pilih armada. Dropdown ini hanya menampilkan armada Anywhere di AWS Region untuk profil pengguna yang sedang aktif.
- Jika tidak ada armada yang ada—atau Anda ingin membuat armada baru, pilih Create new Anywhere fleet dan berikan nama armada.

Setelah Anda memilih armada Anywhere untuk proyek Anda, Amazon GameLift Servers verifikasi bahwa status armada aktif dan menampilkan ID armada. Anda dapat melacak kemajuan permintaan ini di log keluaran editor Unreal.

Langkah 4: Daftarkan workstation Anda

Pada langkah ini, Anda mendaftarkan workstation lokal Anda sebagai sumber daya komputasi di armada Anywhere yang baru.

Untuk mendaftarkan workstation Anda sebagai komputasi Anywhere

1. Masukkan nama komputasi untuk mesin lokal Anda. Jika Anda menambahkan lebih dari satu komputasi dalam armada, nama harus unik.
2. Berikan alamat IP untuk mesin lokal Anda. Bidang ini default ke alamat IP publik mesin Anda. Anda juga dapat menggunakan localhost (127.0.0.1) selama Anda menjalankan klien dan server game Anda di mesin yang sama.
3. Pilih Daftarkan komputasi. Anda dapat melacak kemajuan permintaan ini di log keluaran editor Unreal.

Menanggapi tindakan ini, Amazon GameLift Servers memverifikasi bahwa itu dapat terhubung ke komputasi dan mengembalikan informasi tentang komputasi yang baru terdaftar. Ini juga menciptakan argumen konsol yang dibutuhkan executable game Anda saat menginisialisasi komunikasi dengan layanan. Amazon GameLift Servers

Langkah 5: Hasilkan token autentikasi

Proses server game yang berjalan pada komputasi Anywhere Anda memerlukan token otentikasi untuk melakukan panggilan ke layanan. Amazon GameLift Servers Plugin secara otomatis menghasilkan dan menyimpan token autentikasi untuk armada Anywhere setiap kali Anda meluncurkan server game dari plugin. Nilai token autentikasi disimpan sebagai argumen baris perintah, yang dapat diambil oleh kode server Anda saat runtime.

Contoh kode yang diberikan di atas juga memungkinkan Anda menggunakan [AWS Signature Version 4 \(SigV4\) untuk permintaan API](#). SigV4 adalah protokol AWS penandatanganan untuk menambahkan informasi otentikasi ke permintaan API.

Anda tidak perlu mengambil tindakan apa pun dalam langkah ini.

Langkah 6: Luncurkan game

Pada titik ini, Anda telah menyelesaikan semua tugas yang diperlukan untuk meluncurkan dan memainkan game multipemain Anda di workstation lokal menggunakan Amazon GameLift Servers.

Untuk memainkan game yang Anda host

1. Luncurkan server game Anda. Server game akan memberi tahu Amazon GameLift Servers ketika siap untuk menyelenggarakan sesi game.
2. Luncurkan klien game Anda dan gunakan fungsionalitas baru untuk memulai sesi permainan baru. Permintaan ini dikirim Amazon GameLift Servers melalui layanan backend baru. Sebagai tanggapan, Amazon GameLift Servers, panggil server game, berjalan di mesin lokal Anda, untuk memulai sesi permainan baru. Ketika sesi permainan siap menerima pemain, Amazon GameLift Servers berikan informasi koneksi untuk klien game untuk bergabung dengan sesi permainan.

Plugin untuk Unreal: Menyebarkan game Anda ke armada yang dikelola EC2

Dalam alur kerja ini, terapkan game Anda untuk hosting di sumber daya komputasi berbasis cloud yang dikelola oleh Amazon GameLift Servers. Unggah build server game terintegrasi Anda ke Amazon GameLift Servers layanan untuk penerapan. Jika Anda belum mengintegrasikan kode game Anda, lihat [Plugin untuk Unreal: Integrasikan kode game Anda](#). Ketika alur kerja ini selesai, Anda akan memiliki klien game yang berfungsi yang dapat terhubung ke server game Anda di cloud.

Untuk memulai EC2 alur kerja Amazon yang Amazon GameLift Servers dikelola:

- Di toolbar utama editor Unreal, pilih Amazon GameLift Servers menu, dan pilih Host with Managed. EC2 Tindakan ini membuka halaman plugin Deploy Amazon EC2 Fleet, yang menyajikan proses enam langkah untuk mengintegrasikan, membangun, menyebarkan, dan meluncurkan komponen game Anda.

Langkah 1: Atur profil Anda

Pilih profil yang ingin Anda gunakan saat mengikuti alur kerja ini. Profil yang Anda pilih memengaruhi semua langkah dalam alur kerja. Semua sumber daya yang Anda buat dikaitkan dengan AWS akun profil dan ditempatkan di AWS Wilayah default profil. Izin pengguna profil menentukan akses Anda ke AWS sumber daya dan tindakan.

Untuk mengatur profil pengguna

1. Pilih profil dari daftar dropdown profil yang tersedia. Jika Anda belum memiliki profil atau ingin membuat yang baru, buka GameLift menu Amazon dan pilih Set AWS User Profiles.
2. Jika status bootstrap tidak “Aktif”, pilih profil Bootstrap dan tunggu statusnya berubah menjadi “Aktif”.

Langkah 2: Siapkan kode permainan Anda

Pada langkah ini, siapkan server game dan build klien game Anda untuk bekerja dengan SDK server Amazon GameLift Servers C++ untuk Unreal. Jika Anda belum mengintegrasikan kode game Anda dan membangun klien game dan server yang dapat dieksekusi, lihat. [Plugin untuk Unreal: Integrasikan kode game Anda](#) Masukkan jalur ke executable game Anda di workstation lokal Anda.

Langkah ini dalam alur kerja plugin menyediakan tautan ke instruksi dan kode sumber untuk menyiapkan versi editor Unreal yang dibuat sumber. Anda perlu menggunakan versi yang dibuat sumber saat membangun komponen klien dan server Anda.

Setelah membangun server game yang terintegrasi dengan SDK server, selesaikan tugas-tugas berikut untuk mempersiapkannya untuk diunggah ke Amazon GameLift Servers hosting.

Untuk mempersiapkan build server Anda untuk penyebaran cloud (Windows)

Di `WindowsServer` folder, tempat editor Unreal menyimpan file build server Anda secara default, buat tambahan berikut:

1. Salin skrip instalasi build server ke root **WindowsServer** folder. Skrip penginstalan disertakan dalam unduhan plugin. Cari filenya[project-name]/Plugins/Resources/CloudFormation/extra_server_resources/install.bat. Amazon GameLift Servers menggunakan file ini untuk menginstal server build ke komputasi hosting Anda.
2. Salin **VC_redist.x64.exe** file ke root **WindowsServer** folder. File ini disertakan dalam instalasi Visual Studio Anda. Biasanya terletak di C:/Program Files (x86)/Microsoft Visual Studio/2019/Professional/VC/Redist/MSVC/v142.
3. Tambahkan file library OpenSSL ke build server game Anda.

 Note

Jika server game Anda terintegrasi dengan server SDK versi 5.3.x atau yang lebih baru, Anda dapat melewati langkah ini.

Untuk server game yang terintegrasi dengan server SDK versi 5.2 atau lebih lama, Anda perlu mencari dan menyalin pustaka secara manual. Anda harus menggunakan versi OpenSSL yang sama dengan yang digunakan versi Unreal Engine 5 Anda. Game build yang digunakan dengan library OpenSSL yang salah tidak akan dapat berkomunikasi dengan layanan. Amazon GameLift Servers

- a. Cari pustaka OpenSSL di sumber mesin game Anda. Lokasi bervariasi tergantung pada lingkungan pengembangan Anda:

Di 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

Di 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. Salin pustaka OpenSSL ke direktori paket build game Anda di. <YourGame>/Binaries/Win64

Untuk mempersiapkan build server Anda untuk penyebaran cloud (Linux)

Untuk petunjuk lebih rinci tentang menyiapkan server game yang dibuat untuk Linux, lihat [Membangun SDK server Amazon GameLift Servers untuk Unreal Engine 5 di Amazon Linux](#).

1. Tentukan direktori kerja untuk mengatur file build Anda. Struktur direktori kerja digunakan sebagaimana adanya ke setiap komputasi hosting. Tambahkan server game buatan Linux Anda dan semua file dependen.
2. Buat skrip instalasi build server di root direktori kerja Anda. Jika perlu, buat `install.sh` file dan tambahkan perintah apa pun yang diperlukan untuk menginstal build server game Anda dengan benar. Amazon GameLift Servers menggunakan file ini untuk menginstal server build ke setiap sumber daya EC2 hosting.
3. Tambahkan file library OpenSSL ke build server game Anda.

 Note

Jika server game Anda terintegrasi dengan server SDK versi 5.3.x atau yang lebih baru, Anda dapat melewati langkah ini.

Untuk server game yang terintegrasi dengan server SDK versi 5.2 atau lebih lama, Anda perlu mencari dan menyalin pustaka secara manual. Anda harus menggunakan versi OpenSSL yang sama dengan yang digunakan versi Unreal Engine 5 Anda. Game build yang digunakan dengan library OpenSSL yang salah tidak akan dapat berkomunikasi dengan layanan. Amazon GameLift Servers

- a. Cari pustaka OpenSSL di sumber mesin game Anda. Lokasi bervariasi tergantung pada lingkungan pengembangan Anda:

Di 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`

Di 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. Saat Anda menemukan pustaka OpenSSL, salin pustaka tersebut ke direktori paket build game Anda di. <YourGame>/Binaries/Linux

Langkah 3: Pilih skenario penerapan

Pada langkah ini, Anda memilih solusi hosting game yang ingin Anda terapkan saat ini. Anda dapat memiliki beberapa penerapan game Anda, menggunakan salah satu skenario.

- Armada wilayah tunggal: Menyebarkan server game Anda ke satu armada sumber daya hosting di wilayah default AWS profil aktif. Skenario ini adalah titik awal yang baik untuk menguji integrasi server Anda dengan AWS dan konfigurasi build server. Ini menyebarkan sumber daya berikut:
 - AWS armada (On-Demand) dengan build server game Anda diinstal dan dijalankan.
 - Kumpulan pengguna dan klien Amazon Cognito untuk memungkinkan pemain mengautentikasi dan memulai permainan.
 - Authorizer gateway API yang menautkan kumpulan pengguna dengan APIs.
 - Web ACL untuk membatasi panggilan pemain yang berlebihan ke gateway API.
 - API gateway +Fungsi Lambda bagi pemain untuk meminta slot game. Fungsi ini memanggil `CreateGameSession()` jika tidak ada yang tersedia.
 - API gateway +Lambda berfungsi bagi pemain untuk mendapatkan info koneksi untuk permintaan game mereka.
- FlexMatch armada: Menyebarkan server game Anda ke satu set armada dan menyiapkan mak FlexMatch comblang dengan aturan untuk membuat pertandingan pemain. Skenario ini menggunakan hosting Spot berbiaya rendah dengan struktur multi-armada, multi-lokasi untuk ketersediaan yang tahan lama. Pendekatan ini berguna ketika Anda siap untuk mulai merancang komponen mak comblang untuk solusi hosting Anda. Dalam skenario ini, Anda akan membuat sumber daya dasar untuk solusi ini, yang dapat Anda sesuaikan nanti sesuai kebutuhan. Ini menyebarkan sumber daya berikut:
 - FlexMatch konfigurasi perjodohan dan aturan perjodohan ditetapkan untuk menerima permintaan pemain dan pertandingan formulir.
 - Tiga AWS armada dengan build server game Anda diinstal dan berjalan di beberapa lokasi. Termasuk dua armada Spot dan satu armada On-Demand sebagai cadangan.

- AWS antrian penempatan sesi permainan yang memenuhi permintaan untuk pertandingan yang diusulkan dengan menemukan sumber daya hosting terbaik (berdasarkan kelayakan, biaya, latensi pemain, dll.) Dan memulai sesi permainan.
- Kumpulan pengguna dan klien Amazon Cognito untuk memungkinkan pemain mengautentikasi dan memulai permainan.
- Authorizer gateway API yang menautkan kumpulan pengguna dengan APIs.
- Web API untuk membatasi panggilan pemain yang berlebihan ke gateway API.
- API gateway + Fungsi Lambda bagi pemain untuk meminta slot game. Fungsi ini memanggil `StartMatchmaking()`.
- API gateway + Lambda berfungsi bagi pemain untuk mendapatkan info koneksi untuk permintaan game mereka.
- Tabel Amazon DynamoDB untuk menyimpan tiket perjodohan untuk pemain dan informasi sesi permainan.
- Topik SNS+Lambda berfungsi untuk `GameSessionQueue` menangani acara.

Langkah 4: Tetapkan parameter permainan

Pada langkah ini, Anda menjelaskan game Anda untuk diunggah ke AWS;

- Nama pembuatan server: Berikan nama yang berarti untuk pembuatan server game Anda. AWS menggunakan nama ini untuk merujuk ke salinan build server Anda yang diunggah dan digunakan untuk penerapan.
- Server build OS: Masukkan sistem operasi tempat server Anda dibangun untuk dijalankan. Ini memberi tahu jenis sumber daya komputasi AWS apa yang akan digunakan untuk meng-host game Anda.
- Folder server game: Identifikasi jalur ke folder build server lokal Anda.
- Pembuatan server game: Identifikasi jalur ke server game yang dapat dieksekusi.
- Jalur klien game: Identifikasi jalur ke klien game yang dapat dieksekusi.
- Output konfigurasi klien: Bidang ini perlu menunjuk ke folder di build klien Anda yang berisi AWS konfigurasi Anda. Cari di lokasi berikut: `[client-build]/[project-name]/Content/CloudFormation`.

Langkah 5: Menyebarkan skenario

Pada langkah ini, Anda menerapkan game Anda ke solusi hosting cloud berdasarkan skenario penerapan yang Anda pilih. Proses ini dapat memakan waktu beberapa menit sambil AWS memvalidasi pembuatan server Anda, menyediakan sumber daya hosting, menginstal server game Anda, meluncurkan proses server, dan membuatnya siap untuk meng-host sesi game.

Untuk memulai penerapan, pilih `CloudFormationDeploy`. Anda dapat melacak status hosting game Anda di sini. Untuk informasi selengkapnya, Anda dapat masuk ke konsol AWS Manajemen untuk AWS dan melihat pemberitahuan acara. Pastikan untuk masuk menggunakan akun, pengguna, dan AWS Wilayah yang sama dengan profil pengguna aktif di plugin.

Saat penerapan selesai, server game Anda diinstal pada sebuah AWS EC2 instance. Setidaknya satu proses server berjalan dan siap untuk memulai sesi permainan.

Langkah 6: Luncurkan klien

Pada titik ini, Anda telah menyelesaikan semua tugas yang diperlukan untuk meluncurkan dan memainkan game multipemain Anda yang Amazon GameLift Servers dihosting. Untuk memainkan game, luncurkan instance klien game Anda.

Jika Anda menerapkan skenario armada tunggal, Anda dapat membuka satu instance klien dengan satu pemain, masuk ke peta server dan bergerak. Buka instance tambahan dari klien game untuk menambahkan pemain kedua ke peta game server yang sama.

Jika Anda menerapkan FlexMatch skenario, solusinya menunggu setidaknya dua klien untuk antri untuk penempatan sesi permainan sebelum pemain dapat memasuki peta server.

Plugin untuk Unreal: Menyebarkan game Anda ke armada kontainer terkelola

Gunakan alur kerja plugin terpandu ini untuk membuat gambar kontainer untuk server game Anda dan menerapkannya ke solusi hosting berbasis kontainer. Jika Anda belum mengintegrasikan kode game Anda, lihat [Plugin untuk Unreal: Integrasikan kode game Anda](#). Ketika Anda berhasil menyelesaikan alur kerja ini, server game kontainer Anda berjalan di cloud, dan Anda dapat menggunakan plugin untuk memulai klien game, terhubung ke sesi game, dan memainkan game.

Sebelum Anda mulai

Alur kerja ini mengasumsikan bahwa Anda telah menyelesaikan tugas-tugas berikut.

- Integrasikan kode server game Anda dengan SDK Amazon GameLift Servers server. Server game host Anda harus dapat berkomunikasi dengan Amazon GameLift Servers layanan sehingga dapat menanggapi permintaan untuk memulai sesi permainan baru dan melaporkan status sesi game. Jika Anda belum menyelesaikan tugas ini, kami sarankan Anda mengikuti alur kerja plugin Host with Anywhere terlebih dahulu. Untuk panduan tentang mempersiapkan kode server game Anda, lihat [Perbarui kode server game Anda](#). Untuk armada kontainer terkelola, Anda harus mengintegrasikan game Anda dengan server SDK versi 5.2 atau lebih tinggi.

 Note

Jika Anda menggunakan peta game startup, tugas ini sudah dilakukan untuk Anda.

- Package server game Anda yang dapat dieksekusi untuk berjalan di Linux. [Jika Anda mengembangkan di Windows dan mengintegrasikan SDK server C ++ versi 5.2.x atau sebelumnya, Anda harus bekerja dengan toolkit kompilasi silang Unreal](#). Atau, Anda dapat mengatur ruang kerja Linux terpisah atau menggunakan alat seperti subsistem Windows untuk Linux (WSL).
- Kumpulkan file untuk disebarkan dengan build server game Anda. Di mesin lokal Anda, buat direktori kerja untuk mengatur file, yang akan dibangun ke dalam gambar wadah server game Anda. Ini mungkin termasuk dependensi game, skrip untuk meluncurkan server game dan proses lain saat memulai wadah, dll.
- Tambahkan file library OpenSSL untuk build server game Anda.

 Note

Jika server game Anda terintegrasi dengan server SDK versi 5.3.x atau yang lebih baru, Anda dapat melewati langkah ini.

Untuk server game yang terintegrasi dengan server SDK versi 5.2 atau lebih lama, Anda perlu mencari dan menyalin pustaka secara manual. Anda harus menggunakan versi OpenSSL yang sama dengan yang digunakan versi Unreal Engine 5 Anda. Game build yang digunakan dengan library OpenSSL yang salah tidak akan dapat berkomunikasi dengan layanan Amazon GameLift Servers

- a. Cari pustaka OpenSSL di sumber mesin game Anda. Lokasi bervariasi tergantung pada lingkungan pengembangan Anda:

Di 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

Di 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. Salin pustaka OpenSSL ke direktori paket build game Anda di. <YourGame>/Binaries/Win64

- Integrasikan kode klien game Anda dengan Amazon GameLift Servers. Salah satu cara untuk menyelesaikan tugas ini adalah dengan menambahkan aset sampel (disertakan dengan plugin) yang sudah terintegrasi. Untuk panduan tentang mempersiapkan kode klien game Anda, lihat [Integrasikan peta permainan klien Anda](#).
- Instal Docker di mesin lokal Anda. Anda memerlukan alat ini diinstal jika Anda ingin plugin membuat gambar kontainer untuk Anda dan mendorongnya ke repositori ECR. Atau Anda dapat melakukan tugas-tugas ini secara manual dan menginstruksikan plugin untuk menggunakan gambar kontainer yang ada. Untuk informasi selengkapnya tentang membuat gambar secara manual, lihat [Membuat gambar kontainer untuk Amazon GameLift Servers](#).

Untuk memulai alur kerja kontainer Amazon GameLift Servers terkelola:

- Di toolbar utama editor Unreal, pilih Amazon GameLift Servers menu, dan pilih Kontainer Terkelola. Tindakan ini membuka halaman plugin Host with Managed Containers, yang menyajikan step-by-step proses untuk membuat gambar kontainer dengan build server game Anda, menyebarkannya ke armada kontainer, dan meluncurkan game Anda.

Langkah 0: Atur profil Anda

Bagian ini menampilkan profil pengguna yang Anda pilih saat ini. Verifikasi bahwa profil pengguna saat ini adalah yang ingin Anda gunakan untuk alur kerja ini. Semua sumber daya yang Anda buat dalam alur kerja ini dikaitkan dengan AWS akun profil dan ditempatkan di AWS Wilayah default profil. Izin pengguna profil menentukan akses Anda ke AWS sumber daya dan tindakan.

Anda mungkin perlu mengubah profil pengguna yang dipilih jika:

- Tidak ada profil yang dipilih saat ini.
- Anda ingin memilih profil yang berbeda atau membuat profil baru.
- Anda perlu mem-bootstrap profil yang dipilih (jika status bootstrap tidak aktif).

Untuk mengatur atau mengubah profil pengguna yang dipilih

- Di Amazon GameLift Servers menu, pilih Open AWS Access Credentials.

Langkah 1: Nilai kesiapan wadah

Sebelum menerapkan server game Anda ke armada kontainer, Anda harus mengemasnya ke dalam gambar kontainer dan menyimpannya di repositori Amazon ECR. Plugin dapat menyelesaikan tugas-tugas ini untuk Anda atau Anda dapat melakukan tugas-tugas ini secara manual. Pada langkah ini, berikan informasi tentang status gambar kontainer Anda dan repositori ECR.

Gunakan pertanyaan penilaian untuk memberi tahu plugin langkah-langkah apa yang perlu diambil:

- Buat gambar kontainer baru. Jika Anda memilih opsi ini, langkah selanjutnya akan meminta Anda untuk lokasi direktori build server game Anda dan build yang dapat dieksekusi. Plugin menggunakan template Dockerfile (disediakan oleh Amazon GameLift Servers) dan secara otomatis mengkonfigurasinya untuk game Anda. Anda dapat melihat template di [Membangun gambar kontainer untuk Amazon GameLift Servers](#). Setelah memilih opsi ini, tunjukkan di mana Anda ingin plugin menyimpan gambar baru:
 - Buat repositori Amazon ECR baru dan dorong gambar kontainer ke sana. Plugin membuat repo ECR pribadi menggunakan AWS akun dan default Wilayah AWS di profil pengguna yang Anda pilih.
 - Dorong gambar kontainer ke repositori Amazon ECR yang dibuat sebelumnya. Jika Anda memilih opsi ini, langkah selanjutnya akan meminta Anda untuk memilih repositori Amazon ECR yang ada dari daftar. Daftar ini mencakup semua repositori Amazon ECR untuk AWS akun dan default Wilayah AWS di profil pengguna yang Anda pilih. Anda dapat memilih repositori publik atau pribadi.
- Gunakan gambar kontainer yang ada. Jika Anda telah membuat gambar secara manual, kami sarankan Anda menggunakan template Dockerfile yang disediakan oleh Amazon GameLift Servers, yang tersedia di [Membangun gambar kontainer untuk Amazon GameLift Servers](#). Setelah memilih opsi ini, tunjukkan di mana gambar berada.

- Gambar buatan Docker yang disimpan secara lokal. Jika Anda memilih opsi ini, plugin membuat repositori pribadi Amazon ECR baru dan mendorong file gambar lokal ke sana. Langkah selanjutnya akan meminta Anda untuk ID gambar, yang digunakan plugin untuk menemukan file gambar.
- Gambar kontainer yang sudah disimpan di repositori Amazon ECR. Jika Anda memilih opsi ini, langkah selanjutnya akan meminta Anda untuk memilih repositori Amazon ECR dan gambar yang ada dari daftar. Daftar ini mencakup semua repositori Amazon ECR untuk AWS akun dan default Wilayah AWS di profil pengguna yang Anda pilih. Anda dapat memilih repositori publik atau pribadi.

Langkah 2: Konfigurasi penyebaran gambar

Pada langkah ini, berikan informasi yang diperlukan plugin untuk menyebarkan gambar kontainer Anda ke armada kontainer. Langkah ini meminta informasi berikut:

- Lokasi pembuatan server game, image container, atau repositori Amazon ECR Anda, berdasarkan pilihan Anda di Langkah 1.
- Skenario yang akan digunakan untuk penerapan kontainer terkelola Anda.
- Jalur keluaran konfigurasi klien. Pilih folder di build klien yang berisi AWS konfigurasi Anda. Cari di lokasi berikut: `[client-build]/[project-name]/Content/CloudFormation`.
- Pengaturan penyebaran opsional. Bagian ini memiliki pengaturan konfigurasi yang digunakan plugin secara default. Anda dapat memodifikasi ini atau menyimpan nilai default
 - Nama game diatur ke nama proyek game Anda secara default. Semua AWS sumber daya yang dibuat plugin mereferensikan nilai nama game.
 - Rentang port, batas memori, dan batas vCPU adalah pengaturan konfigurasi untuk armada kontainer. Untuk informasi selengkapnya tentang menyesuaikan nilai ini, lihat [Konfigurasi koneksi jaringan](#) rentang port koneksi, dan [Tetapkan batas sumber daya](#) untuk batas sumber daya.
- Tag gambar kontainer digunakan untuk mengkategorikan gambar kontainer Anda di Amazon ECR. Nilai default-nya adalah `unreal-gamelift-plugin`.
- Nama repositori Amazon ECR. Anda dapat mengedit bidang ini untuk menyarankan nama khusus hanya ketika plugin membuat repositori ECR untuk Anda. Nilai default-nya adalah `unreal-game lift-plugin-ecr-repository`.

Opsi skenario penerapan

Armada kontainer satu wilayah

Skenario ini menyebarkan server game Anda ke armada kontainer tunggal. Ini adalah titik awal yang baik untuk menguji integrasi server Anda dengan AWS dan konfigurasi kontainer Anda. Ini menyebarkan sumber daya berikut.

- Amazon GameLift Servers definisi grup kontainer menjelaskan cara menerapkan dan menjalankan gambar kontainer Anda pada armada kontainer.
- Amazon GameLift Servers armada kontainer (On-Demand) dengan wadah server game Anda diinstal dan dijalankan, dengan alias.
- Kumpulan pengguna dan klien Amazon Cognito untuk memungkinkan pemain mengautentikasi dan memulai permainan.
- Authorizer API Gateway yang menautkan kumpulan pengguna dengan APIs.
- Daftar kontrol akses web (ACL) untuk membatasi panggilan pemain yang berlebihan ke API Gateway.
- Layanan backend untuk membuat permintaan ke Amazon GameLift Servers layanan atas nama klien game, seperti untuk meminta sesi permainan dan bergabung dengan game:
 - API Gateway + Fungsi Lambda bagi pemain untuk meminta slot sesi permainan. Fungsi ini memanggil `CreateGameSession()` jika tidak ada slot terbuka yang tersedia.
 - API Gateway + Fungsi Lambda bagi pemain untuk mendapatkan info koneksi untuk permintaan game mereka.

Armada kontainer satu wilayah dengan FlexMatch

Skenario ini menyebarkan server game Anda ke armada kontainer, mengonfigurasi penempatan sesi game, dan menyiapkan FlexMatch perjodohan. Skenario ini berguna ketika Anda siap untuk mulai merancang mak comblang khusus untuk solusi hosting Anda. Gunakan skenario ini untuk membuat sumber daya dasar untuk solusi ini, yang dapat Anda sesuaikan nanti sesuai kebutuhan. Ini menyebarkan sumber daya berikut:

- Amazon GameLift Servers definisi grup kontainer yang menjelaskan cara menerapkan dan menjalankan gambar kontainer Anda pada armada kontainer.
- Amazon GameLift Servers armada kontainer (On-Demand) dengan wadah server game Anda diinstal dan dijalankan, dengan alias.

- FlexMatch konfigurasi perijinan dan aturan perijinan ditetapkan untuk menerima permintaan pemain dan pertandingan formulir.
- Amazon GameLift Servers antrian sesi permainan yang memenuhi permintaan untuk pertandingan yang diusulkan dengan menemukan sumber daya hosting terbaik (berdasarkan kelayakan, biaya, latensi pemain, dll.) Dan memulai sesi permainan.
- Kumpulan pengguna dan klien Amazon Cognito untuk memungkinkan pemain mengautentikasi dan memulai permainan.
- Authorizer API Gateway yang menautkan kumpulan pengguna dengan APIs.
- Daftar kontrol akses web (ACL) untuk membatasi panggilan pemain yang berlebihan ke API Gateway.
- Layanan backend untuk membuat permintaan ke Amazon GameLift Servers layanan atas nama klien game, seperti untuk meminta sesi permainan dan bergabung dengan game:
 - API Gateway + Fungsi Lambda bagi pemain untuk meminta slot sesi permainan. Fungsi ini memanggil `StartMatchmaking()` jika tidak ada slot terbuka yang tersedia.
 - API Gateway + Fungsi Lambda bagi pemain untuk mendapatkan info koneksi untuk permintaan game mereka.
- Meja DynamoDB untuk menyimpan tiket perijinan untuk pemain dan informasi sesi permainan.
- Topik Amazon SNS+Lambda berfungsi untuk menangani acara. `GameSessionQueue`

Menyebarkan armada kontainer

Ketika konfigurasi armada Anda selesai, pilih tombol `Deploy container fleet` untuk memulai penyebaran. Proses ini dapat memakan waktu beberapa menit saat plugin membuat gambar kontainer dan mendorongnya ke ECR, menyediakan sumber daya hosting untuk armada kontainer, menyebarkan armada dan AWS sumber daya lainnya untuk skenario solusi hosting yang dipilih.

Saat memulai penerapan, Anda dapat melacak kemajuan setiap langkah. Tergantung pada konfigurasi Anda, langkah-langkahnya mungkin termasuk yang berikut:

- Mengkonfigurasi gambar kontainer
- Membuat repositori Amazon ECR
- Membangun gambar dan mendorong ke Amazon ECR
- Membuat definisi grup kontainer
- Membuat armada kontainer

Untuk informasi penerapan yang lebih detail, pilih Lihat di Konsol AWS Manajemen. Ketika armada kontainer mencapai status aktif, armada secara aktif menjalankan kontainer dengan proses server yang siap menjadi tuan rumah sesi permainan.

Saat penyebaran selesai, Anda memiliki armada kontainer yang berfungsi yang siap menyelenggarakan sesi permainan dan menerima koneksi pemain.

Anda tidak dapat menghentikan penerapan yang sedang berlangsung. Jika penerapan memasuki status buruk atau gagal, Anda dapat memulai kembali dengan menggunakan opsi Reset deployment.

Luncurkan klien

Pada titik ini, Anda telah menyelesaikan semua tugas untuk meluncurkan dan memainkan game multipemain Anda yang Amazon GameLift Servers dihosting. Untuk memainkan game Anda, pilih Mulai Klien untuk meluncurkan instance lokal klien game Anda.

- Jika Anda menerapkan skenario armada tunggal, buka satu instance klien game Anda dengan satu pemain dan masukkan peta server untuk bergerak. Anda dapat membuka instance kedua dari klien game untuk menambahkan pemain kedua ke peta game server yang sama.
- Jika Anda menerapkan FlexMatch skenario, solusi hosting menunggu setidaknya dua klien game untuk membuat permintaan perjodohan. Buka setidaknya dua contoh klien game Anda dengan satu pemain. Kedua pemain akan dicocokkan dan diminta untuk bergabung dengan sesi permainan untuk pertandingan.

Perbarui armada kontainer

Jika berhasil menerapkan solusi hosting kontainer terkelola, Anda dapat menggunakan fitur Update deployment. Opsi ini memungkinkan Anda memperbarui pengaturan konfigurasi untuk armada kontainer yang digunakan, tanpa harus membuat armada baru.

Saat memperbarui penerapan, Anda dapat menerapkan image kontainer dengan build server game yang berbeda, mengubah repositori Amazon ECR, memilih skenario penerapan yang berbeda, dan menyesuaikan pengaturan konfigurasi opsional.

Saat Anda siap untuk menerapkan perubahan, pilih Perbarui. Waktu yang diperlukan untuk pembaruan penerapan mirip dengan penerapan penuh. Untuk informasi penyebaran mendetail, pilih Lihat di Konsol AWS Manajemen.

Bersihkan sumber daya yang digunakan

Sebagai praktik terbaik, bersihkan AWS sumber daya untuk solusi kontainer terkelola Anda segera setelah Anda tidak lagi membutuhkannya. Anda mungkin terus mengeluarkan biaya untuk sumber daya ini jika Anda tidak menghapusnya.

Hapus sumber daya berikut:

- Tumpukan sumber daya kontainer terkelola. Sumber daya dalam tumpukan ini bergantung pada skenario penerapan yang Anda pilih. Untuk menghapus seluruh tumpukan, gunakan AWS CloudFormation konsol. Tumpukan yang dihasilkan dari Amazon GameLift Servers plugin menggunakan konvensi penamaan berikut: `GameLiftPluginForUnreal-{GameName}-Containers`. Tunggu proses penghapusan tumpukan selesai sebelum Anda memulai penerapan kontainer terkelola baru di plugin. Untuk informasi selengkapnya, lihat [Menghapus tumpukan dari CloudFormation konsol](#).
- Repositori Amazon ECR. Jika Anda menggunakan plugin untuk membuat repositori untuk gambar kontainer Anda, Anda mungkin ingin menghapus repositori yang tidak lagi diperlukan. Anda tidak perlu menghapus repositori sebelum mengatur ulang penerapan kontainer terkelola. Jika Anda memperbarui atau mengatur ulang penyebaran, plugin akan secara otomatis menggunakan repositori yang sama kecuali diarahkan untuk menggunakan yang lain. Untuk informasi selengkapnya, lihat [Menghapus repositori pribadi di Amazon ECR](#).

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

Plugin ini menambahkan SDK server Amazon GameLift Servers C # dan alat ke editor UE. Gunakan alur kerja UI terpandu untuk mengintegrasikan fungsionalitas SDK server ke dalam proyek game Anda dan menerapkan solusi Amazon GameLift Servers hosting untuk server game Anda.

Dengan plugin, Anda dapat membangun solusi hosting kerja dasar dan kemudian mengoptimalkan dan menyesuaikan sesuai kebutuhan. Siapkan armada Amazon GameLift Servers Anywhere dengan workstation lokal Anda sebagai host. Untuk hosting cloud dengan armada kontainer terkelola EC2 atau terkelola, gunakan server game Anda dengan solusi lengkap untuk mengelola permintaan sesi game dan koneksi klien.

Topik

- [Instal plugin untuk proyek game Unreal Anda](#)
- [Plugin untuk Unity: Siapkan profil AWS pengguna](#)

- [Plugin untuk Unity: Siapkan pengujian lokal dengan Amazon GameLift Servers Anywhere](#)
- [Plugin for Unity: Terapkan game Anda ke armada yang dikelola EC2](#)
- [Plugin for Unity: Terapkan game Anda ke armada kontainer terkelola](#)

Instal plugin untuk proyek game Unreal Anda

[Dapatkan Amazon GameLift Servers plugin untuk Unity dari GitHub](#)

Lihat readme GitHub repositori untuk informasi tentang cara menginstal plugin untuk proyek game.

Plugin ini mencakup komponen-komponen ini:

- Modul plugin untuk editor Unity. Ketika plugin diinstal, item menu utama baru memberi Anda akses ke Amazon GameLift Servers fungsionalitas.
- Pustaka C# untuk API Amazon GameLift Servers layanan dengan fungsionalitas sisi klien.
- Pustaka C# untuk SDK Amazon GameLift Servers server (versi 5.x).
- Cicipi konten game, termasuk aset dan adegan, sehingga Anda dapat mencoba Amazon GameLift Servers bahkan jika Anda tidak memiliki game multipemain yang siap dibangun.
- Konfigurasi solusi, disediakan sebagai AWS CloudFormation templat, yang digunakan plugin saat menyebarkan server game Anda ke cloud untuk hosting.

Plugin ini menggunakan AWS CloudFormation template untuk menyebarkan solusi hosting untuk skenario game umum. Gunakan solusi ini seperti yang disediakan atau sesuaikan sesuai kebutuhan untuk game Anda.

Plugin untuk Unity: Siapkan profil AWS pengguna

Setelah menginstal plugin, atur profil pengguna dengan AWS akun yang valid. Anda dapat mempertahankan beberapa profil di plugin, tetapi Anda hanya dapat memiliki satu profil yang dipilih pada satu waktu. Setiap kali Anda bekerja di plugin, pilih profil yang akan digunakan. Setiap halaman alur kerja menampilkan profil yang dipilih saat ini.

Mempertahankan beberapa profil memberi Anda kemampuan untuk beralih di antara penerapan hosting yang berbeda. Misalnya, Anda dapat mengatur profil yang menggunakan AWS akun yang sama tetapi menyebarkan ke AWS Wilayah yang berbeda. Atau Anda dapat mengatur profil dengan AWS akun atau pengguna yang berbeda dan set izin.

 Note

Jika Anda telah menginstal AWS CLI di workstation Anda dan memiliki profil yang sudah dikonfigurasi, Amazon GameLift Servers plugin akan mendeteksi dan mencantumkannya sebagai profil yang ada. Plugin secara otomatis memilih profil apa pun yang bernama `default`. Anda dapat menggunakan profil yang ada atau membuat yang baru.

Semua profil harus di-bootstrap untuk menyiapkan beberapa AWS sumber daya yang diperlukan di bawah pengguna akun Anda.

Untuk mengelola AWS profil Anda

1. Di toolbar utama Unity, pilih Amazon GameLift Servers menu, dan pilih AWS Access Credentials. Tindakan ini membuka Amazon GameLift Servers plugin ke halaman Siapkan profil pengguna Anda.
2. Gunakan tombol untuk membuat AWS akun baru atau mengatur profil pengguna untuk AWS akun yang sudah Anda miliki.
3. Jika Anda belum memiliki profil pengguna, Anda diminta untuk memasukkan detail profil dan membuat profil baru. Saat diminta, berikan informasi berikut:
 - Sebuah AWS akun. Jika Anda membuat AWS akun baru, gunakan tautan ke AWS Management Console dan ikuti petunjuknya. Lihat [Membuat AWS akun](#) untuk detail selengkapnya.
 - AWS Pengguna dengan izin untuk menggunakan Amazon GameLift Servers dan AWS layanan lain yang diperlukan. Lihat [Mengatur sebuah Akun AWS](#) petunjuk tentang cara menyiapkan pengguna AWS Identity and Access Management (IAM) dengan Amazon GameLift Servers izin dan akses terprogram dengan kredensial jangka panjang.
 - Kredensi untuk pengguna Anda AWS . Kredensial ini terdiri dari ID kunci AWS akses dan kunci AWS rahasia. Lihat [Dapatkan kunci akses Anda](#) untuk detail selengkapnya.
 - AWS wilayah. Ini adalah lokasi geografis tempat Anda ingin membuat AWS sumber daya untuk hosting. Selama pengembangan, kami sarankan menggunakan wilayah yang dekat dengan lokasi fisik Anda. Pilih wilayah dari daftar [AWS wilayah yang didukung](#).
4. Jika plugin mendeteksi profil yang ada, itu akan menampilkan daftar profil yang tersedia. Pilih profil yang ada dari daftar, atau pilih Tambahkan profil lain untuk membuat profil baru.

Bootstrap profil pengguna

Semua profil harus di-bootstrap untuk digunakan dengan plugin. Amazon GameLift Servers Bootstrapping membuat bucket Amazon S3 khusus untuk profil. Ini digunakan untuk menyimpan konfigurasi proyek, membangun artefak, dan dependensi lainnya. Bucket tidak dibagikan di antara profil lain.

Bootstrapping melibatkan pembuatan AWS sumber daya baru dan mungkin menimbulkan biaya.

Untuk bootstrap profil Anda:

1. Pada halaman AWS Access Credentials, periksa status bootstrap dari profil pengguna yang ingin Anda gunakan. Jika status bootstrap profil adalah “Tidak Aktif” dan tidak ada ember S3 yang terdaftar, Anda perlu mem-bootstrap profil.
2. Pilih profil yang ingin Anda gunakan dan pilih profil Bootstrap.
3. Tunggu status bootstrap berubah menjadi “Aktif”. Hal ini dapat menghabiskan waktu beberapa menit.

Plugin untuk Unity: Siapkan pengujian lokal dengan Amazon GameLift Servers Anywhere

Dalam alur kerja ini, Anda menambahkan kode game klien dan server untuk Amazon GameLift Servers fungsionalitas dan menggunakan plugin untuk menunjuk workstation lokal Anda sebagai host server game uji. Ketika Anda telah menyelesaikan tugas integrasi, gunakan plugin untuk membangun klien game dan komponen server Anda.

Untuk memulai alur kerja Amazon GameLift Servers Anywhere:

- Di menu utama editor Unity, pilih Amazon GameLift Servers dan pilih Host with Anywhere. Tindakan ini membuka halaman plugin untuk menyiapkan game Anda dengan armada @Anywhere. Halaman ini menyajikan proses lima langkah untuk mengintegrasikan, membangun, dan meluncurkan komponen game Anda.

Tetapkan profil Anda

Pilih profil yang ingin Anda gunakan saat mengikuti alur kerja ini. Profil yang Anda pilih memengaruhi semua langkah dalam alur kerja. Semua sumber daya yang Anda buat dikaitkan dengan AWS akun

profil dan ditempatkan di AWS Wilayah default profil. Izin pengguna profil menentukan akses Anda ke AWS sumber daya dan tindakan.

1. Pilih profil dari daftar dropdown profil yang tersedia. Jika Anda belum memiliki profil atau ingin membuat yang baru, buka Amazon GameLift Servers menu dan pilih Set AWS Account Profiles.
2. Jika status bootstrap bukan “Aktif”, pilih profil Bootstrap dan tunggu statusnya berubah menjadi “Aktif”.

Integrasikan kode game Anda dengan SDK server C #

 Note

Jika Anda mengimpor contoh permainan, Anda dapat melewati langkah ini. Aset permainan sampel sudah memiliki server dan kode klien yang diperlukan.

Untuk langkah ini dalam alur kerja, Anda membuat pembaruan untuk klien dan kode server dalam proyek game Anda.

- * Server game harus dapat berkomunikasi dengan Amazon GameLift Servers layanan untuk menerima petunjuk untuk memulai sesi permainan, memberikan informasi koneksi sesi game, dan melaporkan status.
- Klien game harus bisa mendapatkan informasi tentang sesi permainan, bergabung atau memulai sesi permainan, dan mendapatkan informasi koneksi untuk bergabung dengan game.

Integrasikan kode server Anda

Jika Anda menggunakan proyek game Anda sendiri dengan adegan khusus, gunakan kode sampel yang disediakan untuk menambahkan kode server yang diperlukan ke proyek game Anda:

1. Dalam file proyek game Anda, buka Assets/Scripts/Server folder. Jika tidak ada, buatlah.
2. Pergi ke GitHub repo [aws/ amazon-gamelift-plugin-unity](#) dan buka jalurnya. Samples~/SampleGame/Assets/Scripts/Server
3. Temukan file `GameLiftServer.cs` dan salin ke Server folder proyek game Anda. Saat Anda membangun server yang dapat dieksekusi, gunakan file ini sebagai target build.

Kode sampel mencakup elemen minimum yang diperlukan ini, yang menggunakan SDK server Amazon GameLift Servers C # (versi 5):

- Menginisialisasi klien Amazon GameLift Servers API. `InitSDK()` Panggilan dengan parameter server diperlukan untuk armada Amazon GameLift Servers Anywhere. Pengaturan ini secara otomatis diatur untuk digunakan dalam plugin.
- Menerapkan fungsi callback yang diperlukan untuk menanggapi permintaan dari Amazon GameLift Servers layanan, termasuk, `OnStartGameSessionOnProcessTerminate`, dan `onHealthCheck`
- Panggilan `ProcessReady()` dengan port yang ditunjuk untuk memberi tahu Amazon GameLift Servers layanan ketika proses server siap untuk meng-host sesi permainan.

Jika Anda ingin menyesuaikan kode server sampel, lihat sumber daya ini:

- [Tambahkan Amazon GameLift Servers ke server game Anda](#)
- [C# server SDK 5.x untuk Amazon GameLift Servers -- Tindakan](#)

Integrasikan kode klien Anda

Jika Anda menggunakan proyek game Anda sendiri dengan adegan khusus, maka Anda perlu mengintegrasikan fungsionalitas dasar ke dalam klien game Anda. Anda juga perlu menambahkan elemen UI sehingga pemain dapat masuk dan bergabung dengan sesi permainan. Gunakan API layanan untuk Amazon GameLift Servers (dalam AWS SDK) untuk mendapatkan informasi sesi game, membuat sesi game baru, atau bergabung dengan sesi game yang ada,

Saat membangun klien untuk pengujian lokal dengan armada Anywhere, Anda dapat menambahkan panggilan langsung ke Amazon GameLift Servers layanan. Saat Anda mengembangkan game untuk cloud hosting—atau jika Anda berencana menggunakan armada Anywhere untuk hosting produksi—Anda harus membuat layanan backend sisi klien untuk menangani semua komunikasi antara klien game dan layanan. Amazon GameLift Servers

Untuk mengintegrasikan Amazon GameLift Servers ke dalam kode klien Anda, gunakan sumber daya berikut sebagai panduan.

- Integrasikan klien dengan `GameLiftCoreApi` kelas di GitHub repo `amazon-gamelift-plugin-unity-aws/`. Kelas ini menyediakan kontrol untuk otentikasi pemain dan untuk mengambil informasi sesi permainan.

- Lihat contoh integrasi game, tersedia di GitHub repo `amazon-gamelift-plugin-unity aws/. Samples~/SampleGame/Assets/Scripts/Client/GameLiftClient.cs`
- Ikuti petunjuk di Tambahkan Amazon GameLift Servers ke klien game Unity Anda.

Untuk klien game yang terhubung ke armada Anywhere, klien game Anda membutuhkan informasi berikut. Plugin secara otomatis memperbarui proyek game Anda untuk menggunakan sumber daya yang Anda buat di plugin.

- `FleetId` - Pengidentifikasi unik untuk armada Anywhere Anda.
- `FleetLocation` - Lokasi kustom armada Anywhere Anda.
- `AwsRegion` - AWS Wilayah tempat armada Anywhere Anda dihosting. Ini adalah wilayah yang Anda tetapkan di profil pengguna Anda.
- `ProfileName` - Profil AWS kredensial di mesin lokal Anda yang memungkinkan akses ke AWS SDK untuk. Amazon GameLift Servers Klien game menggunakan kredensial ini untuk mengautentikasi permintaan ke layanan. Amazon GameLift Servers

Note

Profil kredensial dihasilkan oleh plugin dan disimpan di mesin lokal. Akibatnya, Anda harus menjalankan klien di mesin lokal (atau pada mesin dengan profil yang sama).

Connect ke armada Anywhere

Pada langkah ini, Anda menunjuk armada Anywhere untuk digunakan. Armada Anywhere mendefinisikan kumpulan sumber daya komputasi, yang dapat ditemukan di mana saja, untuk hosting server game.

- Jika AWS akun yang saat ini Anda gunakan memiliki armada Anywhere yang sudah ada, buka bidang tarik-turun nama Armada dan pilih armada. Dropdown ini hanya menampilkan armada Anywhere di AWS Region untuk profil pengguna yang sedang aktif.
- Jika tidak ada armada yang ada—atau Anda ingin membuat armada baru, pilih `Create new Anywhere fleet` dan berikan nama armada.

Setelah Anda memilih armada Anywhere untuk proyek Anda, Amazon GameLift Servers verifikasi bahwa status armada aktif dan menampilkan ID armada. Anda dapat melacak kemajuan permintaan ini di log keluaran editor Unity.

Daftarkan komputasi

Pada langkah ini, Anda mendaftarkan workstation lokal Anda sebagai sumber daya komputasi di armada Anywhere yang baru.

1. Masukkan nama komputasi untuk mesin lokal Anda. Jika Anda menambahkan lebih dari satu komputasi dalam armada, nama harus unik.
2. Pilih Daftarkan komputasi. Anda dapat melacak kemajuan permintaan ini di log keluaran editor Unity.

Plugin mendaftarkan workstation lokal Anda dengan alamat IP yang disetel ke localhost (127.0.0.1). Pengaturan ini mengasumsikan bahwa Anda akan menjalankan klien dan server game Anda di mesin yang sama.

Menanggapi tindakan ini, Amazon GameLift Servers memverifikasi bahwa itu dapat terhubung ke komputasi dan mengembalikan informasi tentang komputasi yang baru terdaftar.

Luncurkan game

Pada langkah ini Anda membangun komponen game Anda dan meluncurkannya untuk memainkan game. Lakukan hal-hal berikut:

1. Konfigurasi klien game Anda. Pada langkah ini, Anda meminta plugin untuk memperbarui `GameLiftClientSettings` aset untuk proyek game Anda. Plugin menggunakan aset ini untuk menyimpan informasi tertentu yang dibutuhkan klien game Anda untuk terhubung ke Amazon GameLift Servers layanan.
 - a. Jika Anda tidak mengimpor dan menginisialisasi game sampel, buat `GameLiftClientSettings` aset baru. Di menu utama editor Unity, pilih Aset, Buat, Amazon GameLift, Pengaturan Klien. Jika Anda membuat beberapa salinan `GameLiftClientSettings` dalam proyek Anda, plugin secara otomatis mendeteksi ini dan memberi tahu Anda aset mana yang akan diperbarui plugin.
 - b. Di Luncurkan Game, pilih Konfigurasi Klien: Terapkan Pengaturan Di Mana Saja. Tindakan ini memperbarui pengaturan klien game Anda untuk menggunakan armada Anywhere yang baru saja Anda atur.

2. Bangun dan jalankan klien game Anda.
 - a. Bangun klien yang dapat dieksekusi menggunakan proses pembuatan Unity standar. Di File, Build Settings, alihkan platform ke Windows, Mac, Linux. Jika Anda mengimpor contoh game dan menginisialisasi pengaturan, daftar build dan target build akan diperbarui secara otomatis.
 - b. Luncurkan satu atau beberapa contoh klien game yang baru dibangun yang dapat dieksekusi.
3. Luncurkan server game di armada Anywhere Anda. Pilih Server: Luncurkan Server di Editor. Tugas ini memulai server langsung yang dapat dihubungkan oleh klien Anda selama editor Unity tetap terbuka.
4. Mulai atau bergabunglah dengan sesi permainan. Dalam instance klien game Anda, gunakan UI untuk bergabung dengan setiap klien ke sesi game. Bagaimana Anda melakukan ini tergantung pada bagaimana Anda menambahkan fungsionalitas ke klien.

Jika Anda menggunakan klien game sampel, ia memiliki karakteristik sebagai berikut:

- Komponen login pemain. Saat menghubungkan ke server game di armada Anywhere, tidak ada validasi pemain. Anda dapat memasukkan nilai apa pun untuk bergabung dengan sesi permainan.
- UI game gabungan sederhana. Ketika klien mencoba untuk bergabung dengan permainan, klien secara otomatis mencari sesi permainan aktif dengan slot pemain yang tersedia. Jika tidak ada sesi permainan yang tersedia, klien meminta sesi permainan baru. Jika sesi permainan tersedia, klien meminta untuk bergabung dengan sesi permainan yang tersedia. Saat menguji game Anda dengan beberapa klien bersamaan, klien pertama memulai sesi permainan, dan klien yang tersisa secara otomatis bergabung dengan sesi permainan yang ada.
- Sesi permainan dengan empat slot pemain. Anda dapat meluncurkan hingga empat instance klien game secara bersamaan dan mereka akan bergabung dengan sesi game yang sama.

Luncurkan dari server yang dapat dieksekusi (opsional)

Anda dapat membangun dan meluncurkan server game yang dapat dieksekusi untuk pengujian pada armada Anywhere.

1. Bangun server yang dapat dieksekusi menggunakan proses pembuatan Unity standar. Di File, Build Settings, alihkan platform ke Dedicated Server dan build.

2. Dapatkan token otentikasi jangka pendek dengan memanggil [get-compute-auth-token](#) perintah AWS CLI dengan ID AWS armada dan Wilayah Anywhere Anda. ID armada ditampilkan di Connect to Anywhere Fleet saat Anda membuat armada. AWS Wilayah ditampilkan di Atur Profil Anda saat Anda memilih profil aktif Anda.

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

3. Luncurkan server game yang baru dibangun yang dapat dieksekusi dari baris perintah dan teruskan token autentikasi yang valid.

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

Plugin for Unity: Terapkan game Anda ke armada yang dikelola EC2

Dalam alur kerja ini, Anda menggunakan plugin untuk mempersiapkan game Anda untuk hosting pada sumber daya komputasi berbasis cloud yang dikelola oleh Amazon GameLift Servers Anda menambahkan kode permainan klien dan server untuk Amazon GameLift Servers fungsionalitas, lalu unggah build server Anda ke Amazon GameLift Servers layanan untuk hosting. Ketika alur kerja ini selesai, Anda akan memiliki server game yang berjalan di cloud dan klien game yang berfungsi yang dapat terhubung dengannya.

Untuk memulai EC2 alur kerja Amazon yang Amazon GameLift Servers dikelola:

- Di menu utama editor Unity, pilih Amazon GameLift Servers dan pilih Host with Managed EC2. Alur kerja ini menyajikan proses enam langkah untuk mengintegrasikan, membangun, menyebarkan, dan meluncurkan komponen game Anda.

Tetapkan profil Anda

Pilih profil yang ingin Anda gunakan saat mengikuti alur kerja ini. Profil yang Anda pilih memengaruhi semua langkah dalam alur kerja. Semua sumber daya yang Anda buat dikaitkan dengan AWS akun profil dan ditempatkan di AWS Wilayah default profil. Izin pengguna profil menentukan akses Anda ke AWS sumber daya dan tindakan.

1. Pilih profil dari daftar dropdown profil yang tersedia. Jika Anda belum memiliki profil atau ingin membuat yang baru, buka Amazon GameLift Servers menu dan pilih Set AWS Account Profiles.

2. Jika status bootstrap bukan “Aktif”, pilih profil Bootstrap dan tunggu statusnya berubah menjadi “Aktif”.

Integrasikan game Anda dengan Amazon GameLift Servers

Untuk tugas ini, Anda membuat pembaruan untuk klien dan kode server dalam proyek game Anda.

- Server game harus dapat berkomunikasi dengan Amazon GameLift Servers layanan untuk menerima petunjuk untuk memulai sesi permainan, memberikan informasi koneksi sesi game, dan melaporkan status.
- Klien game harus bisa mendapatkan informasi tentang sesi permainan, bergabung atau memulai sesi permainan, dan mendapatkan informasi koneksi untuk bergabung dengan game.

Note

Jika Anda mengimpor contoh permainan, Anda dapat melewati langkah ini. Aset permainan sampel sudah memiliki server dan kode klien yang diperlukan.

Integrasikan kode server Anda

Saat menggunakan proyek game Anda sendiri dengan adegan khusus, gunakan kode sampel yang disediakan untuk menambahkan kode server yang diperlukan ke proyek game Anda. Jika Anda mengintegrasikan proyek game Anda untuk pengujian dengan armada Anywhere, Anda telah menyelesaikan instruksi dalam langkah ini.

1. Dalam file proyek game Anda, buka `Assets/Scripts/Server` folder. Jika tidak ada, buatlah.
2. Pergi ke GitHub repo [aws/ amazon-gamelift-plugin-unity](https://github.com/aws/amazon-gamelift-plugin-unity) dan buka jalurnya. `Samples~/SampleGame/Assets/Scripts/Server`
3. Temukan file `GameLiftServer.cs` dan salin ke `Server` folder proyek game Anda. Saat Anda membangun server yang dapat dieksekusi, gunakan file ini sebagai target build.

Kode sampel mencakup elemen minimum yang diperlukan ini, yang menggunakan SDK server Amazon GameLift Servers C # (versi 5):

- Menginisialisasi klien Amazon GameLift Servers API. Panggilan `initSDK()` dengan parameter server diperlukan untuk armada Anywhere. Amazon GameLift Servers Pengaturan ini secara otomatis diatur untuk digunakan dalam plugin.
- Menerapkan fungsi callback yang diperlukan untuk menanggapi permintaan dari Amazon GameLift Servers layanan, termasuk, `OnStartGameSessionOnProcessTerminate`, dan `onHealthCheck`
- Panggilan `ProcessReady()` dengan port yang ditunjuk untuk memberi tahu Amazon GameLift Servers layanan ketika proses server siap untuk meng-host sesi permainan.

Jika Anda ingin menyesuaikan kode server sampel, lihat sumber daya ini:

- [Tambahkan Amazon GameLift Servers ke server game Anda](#)
- [C# server SDK 5.x untuk Amazon GameLift Servers -- Tindakan](#)

Integrasikan kode klien Anda

Untuk klien game yang terhubung ke server game berbasis cloud, sebaiknya gunakan layanan backend sisi klien untuk melakukan panggilan ke Amazon GameLift Servers layanan, alih-alih melakukan panggilan langsung dari klien game.

Dalam alur kerja plugin untuk hosting pada EC2 armada terkelola, setiap skenario penerapan menyertakan layanan backend bawaan yang mencakup komponen berikut:

- Satu set fungsi Lambda dan tabel DynamoDB yang digunakan untuk meminta sesi permainan dan mengambil informasi sesi permainan. Komponen ini menggunakan gateway API sebagai proxy.
- Kumpulan pengguna Amazon Cognito yang menghasilkan pemutar unik IDs dan mengautentikasi koneksi pemain.

Untuk menggunakan komponen ini, klien game Anda memerlukan fungsionalitas untuk mengirim permintaan ke layanan backend untuk melakukan hal berikut:

- Buat pengguna pemain di kumpulan pengguna AWS Cognito dan autentikasi.
- Bergabunglah dengan sesi permainan dan terima informasi koneksi.
- Bergabunglah dengan permainan menggunakan perjodohan.

Gunakan sumber daya berikut sebagai panduan.

- Integrasikan klien dengan [GameLiftCoreApi](#) kelas di GitHub repo [amazon-gamelift-plugin-unityaws/](#). Kelas ini menyediakan kontrol untuk otentikasi pemain dan untuk mengambil informasi sesi permainan.
- Untuk melihat contoh integrasi game, buka GitHub repo [amazon-gamelift-plugin-unityaws/](#),
`Samples~/SampleGame/Assets/Scripts/Client/GameLiftClient.cs`
- [Tambahkan Amazon GameLift Servers ke klien game Anda.](#)

Pilih skenario penerapan

Pada langkah ini, Anda memilih solusi hosting game yang ingin Anda terapkan saat ini. Anda dapat memiliki beberapa penerapan game Anda, menggunakan salah satu skenario.

- Armada wilayah tunggal: Menyebarkan server game Anda ke satu armada sumber daya hosting di wilayah default AWS profil aktif. Skenario ini adalah titik awal yang baik untuk menguji integrasi server Anda dengan AWS dan konfigurasi build server. Ini menyebarkan sumber daya berikut:
 - AWS armada (On-Demand) dengan build server game Anda diinstal dan dijalankan.
 - Kumpulan pengguna dan klien Amazon Cognito untuk memungkinkan pemain mengautentikasi dan memulai permainan.
 - Authorizer gateway API yang menautkan kumpulan pengguna dengan APIs.
 - Web ACL untuk membatasi panggilan pemain yang berlebihan ke gateway API.
 - API gateway + Fungsi Lambda bagi pemain untuk meminta slot game. Fungsi ini memanggil `CreateGameSession()` jika tidak ada yang tersedia.
 - API gateway + Lambda berfungsi bagi pemain untuk mendapatkan info koneksi untuk permintaan game mereka.
- FlexMatch armada: Menyebarkan server game Anda ke satu set armada dan menyiapkan mak FlexMatch comblang dengan aturan untuk membuat pertandingan pemain. Skenario ini menggunakan hosting Spot berbiaya rendah dengan struktur multi-armada, multi-lokasi untuk ketersediaan yang tahan lama. Pendekatan ini berguna ketika Anda siap untuk mulai merancang komponen mak comblang untuk solusi hosting Anda. Dalam skenario ini, Anda akan membuat sumber daya dasar untuk solusi ini, yang dapat Anda sesuaikan nanti sesuai kebutuhan. Ini menyebarkan sumber daya berikut:
 - FlexMatch konfigurasi perjodohan dan aturan perjodohan ditetapkan untuk menerima permintaan pemain dan pertandingan formulir.
 - Tiga AWS armada dengan build server game Anda diinstal dan berjalan di beberapa lokasi. Termasuk dua armada Spot dan satu armada On-Demand sebagai cadangan.

- AWS antrian penempatan sesi permainan yang memenuhi permintaan untuk pertandingan yang diusulkan dengan menemukan sumber daya hosting terbaik (berdasarkan kelayakan, biaya, latensi pemain, dll.) Dan memulai sesi permainan.
- Kumpulan pengguna dan klien Amazon Cognito untuk memungkinkan pemain mengautentikasi dan memulai permainan.
- Authorizer gateway API yang menautkan kumpulan pengguna dengan APIs.
- Web ACL untuk membatasi panggilan pemain yang berlebihan ke gateway API.
- API gateway +Fungsi Lambda bagi pemain untuk meminta slot game. Fungsi ini memanggil `StartMatchmaking()`.
- API gateway +Lambda berfungsi bagi pemain untuk mendapatkan info koneksi untuk permintaan game mereka.
- Tabel Amazon DynamoDB untuk menyimpan tiket perjodohan untuk pemain dan informasi sesi permainan.
- Topik SNS+Lambda berfungsi untuk `GameSessionQueue` menangani acara.

Tetapkan parameter permainan

Pada langkah ini, Anda menjelaskan game Anda untuk diunggah. AWS

- Nama game: Berikan nama yang berarti untuk proyek game Anda. Nama ini digunakan dalam plugin.
- Nama armada: Berikan nama yang berarti untuk EC2 armada terkelola Anda. Amazon GameLift Servers menggunakan nama ini (bersama dengan ID armada) saat mencantumkan sumber daya di AWS konsol.
- Nama build: Berikan nama yang berarti untuk build server Anda. AWS menggunakan nama ini untuk merujuk ke salinan build server Anda yang diunggah Amazon GameLift Servers dan digunakan untuk penerapan.
- Parameter peluncuran: Masukkan instruksi opsional untuk dijalankan saat meluncurkan server yang dapat dieksekusi pada instance EC2 armada terkelola. Panjang maksimum adalah 1024 karakter.
- Folder server game: Berikan jalur ke folder lokal yang berisi build server Anda.
- File server game: Tentukan nama file server yang dapat dieksekusi.

Menyebarkan skenario

Pada langkah ini, Anda menerapkan game Anda ke solusi hosting cloud berdasarkan skenario penerapan yang Anda pilih. Proses ini dapat memakan waktu beberapa menit sambil AWS memvalidasi pembuatan server Anda, menyediakan sumber daya hosting, menginstal server game Anda, meluncurkan proses server, dan membuatnya siap untuk meng-host sesi game.

Untuk memulai penerapan, pilih `CloudFormationDeploy`. Anda dapat melacak status hosting game Anda di sini. Untuk informasi selengkapnya, Anda dapat masuk ke konsol AWS Manajemen untuk AWS dan melihat pemberitahuan acara. Pastikan untuk masuk menggunakan akun, pengguna, dan AWS Wilayah yang sama dengan profil pengguna aktif di plugin.

Saat penerapan selesai, server game Anda diinstal pada sebuah AWS EC2 instance. Setidaknya satu proses server berjalan dan siap untuk memulai sesi permainan.

Luncurkan klien game

Ketika armada Anda berhasil digunakan, Anda sekarang memiliki server game yang berjalan dan tersedia untuk menyelenggarakan sesi game. Anda sekarang dapat membangun klien Anda, meluncurkannya, terhubung untuk bergabung dengan sesi permainan.

1. Konfigurasi klien game Anda. Pada langkah ini, Anda meminta plugin untuk memperbarui `GameLiftClientSettings` aset untuk proyek game Anda. Plugin menggunakan aset ini untuk menyimpan informasi tertentu yang dibutuhkan klien game Anda untuk terhubung ke Amazon GameLift Servers layanan.
 - a. Jika Anda tidak mengimpor dan menginisialisasi game sampel, buat `GameLiftClientSettings` aset baru. Di menu utama editor Unity, pilih Aset, Buat, Amazon GameLift, Pengaturan Klien. Jika Anda membuat beberapa salinan `GameLiftClientSettings` dalam proyek Anda, plugin secara otomatis mendeteksi ini dan memberi tahu Anda aset mana yang akan diperbarui plugin.
 - b. Di Luncurkan Game, pilih Konfigurasi Klien: Terapkan EC2 Pengaturan Terkelola. Tindakan ini memperbarui pengaturan klien game Anda untuk menggunakan EC2 armada terkelola yang baru saja Anda gunakan.
2. Bangun klien game Anda. Bangun klien yang dapat dieksekusi menggunakan proses pembuatan Unity standar. Di File, Build Settings, alihkan platform ke Windows, Mac, Linux. Jika Anda mengimpor contoh game dan menginisialisasi pengaturan, daftar build dan target build akan diperbarui secara otomatis.

3. Luncurkan klien game build baru yang dapat dieksekusi. Untuk mulai bermain game, mulai dua hingga empat instance klien dan gunakan UI di masing-masing untuk bergabung dengan sesi permainan.

Jika Anda menggunakan klien game sampel, ia memiliki karakteristik sebagai berikut:

- Komponen login pemain. Saat menghubungkan ke server game di armada Anywhere, tidak ada validasi pemain. Anda dapat memasukkan nilai apa pun untuk bergabung dengan sesi permainan.
- UI game gabungan sederhana. Ketika klien mencoba untuk bergabung dengan permainan, klien secara otomatis mencari sesi permainan aktif dengan slot pemain yang tersedia. Jika tidak ada sesi permainan yang tersedia, klien meminta sesi permainan baru. Jika sesi permainan tersedia, klien meminta untuk bergabung dengan sesi permainan yang tersedia. Saat menguji game Anda dengan beberapa klien bersamaan, klien pertama memulai sesi permainan, dan klien yang tersisa secara otomatis bergabung dengan sesi permainan yang ada.
- Sesi permainan dengan empat slot pemain. Anda dapat meluncurkan hingga empat instance klien game secara bersamaan dan mereka akan bergabung dengan sesi game yang sama.

Plugin for Unity: Terapkan game Anda ke armada kontainer terkelola

Gunakan alur kerja plugin terpandu ini untuk membuat gambar kontainer untuk server game Anda dan menerapkannya ke solusi hosting berbasis kontainer. Ketika Anda berhasil menyelesaikan alur kerja ini, server game kontainer Anda berjalan di cloud, dan Anda dapat menggunakan plugin untuk memulai klien game, terhubung ke sesi game, dan memainkan game.

Sebelum Anda mulai

Alur kerja ini mengasumsikan bahwa Anda telah menyelesaikan tugas-tugas berikut.

- Integrasikan kode server game Anda dengan SDK Amazon GameLift Servers server. server game host kami harus dapat berkomunikasi dengan Amazon GameLift Servers layanan sehingga dapat menanggapi permintaan untuk memulai sesi permainan baru dan melaporkan status sesi game. Jika Anda belum menyelesaikan tugas ini, kami sarankan Anda mengikuti alur kerja plugin Host with Anywhere terlebih dahulu. Untuk panduan tentang mempersiapkan kode server game Anda, lihat [Integrasikan kode server Anda](#). Untuk armada kontainer terkelola, Anda harus mengintegrasikan game Anda dengan server SDK versi 5.2 atau lebih tinggi.

Note

Jika Anda menggunakan peta game startup, tugas ini sudah dilakukan untuk Anda.

- Package server game Anda yang dapat dieksekusi untuk berjalan di Linux.
- Kumpulkan file untuk disebar dengan build server game Anda. Di mesin lokal Anda, buat direktori kerja untuk mengatur file, yang akan dibangun ke dalam gambar wadah server game Anda. Ini mungkin termasuk dependensi game, skrip untuk meluncurkan server game dan proses lain saat memulai wadah, dll.
- Integrasikan kode klien game Anda dengan Amazon GameLift Servers. Salah satu cara untuk menyelesaikan tugas ini adalah dengan menambahkan aset sampel (disertakan dengan plugin) yang sudah terintegrasi. Untuk panduan tentang mempersiapkan kode klien game Anda, lihat [Integrasikan kode klien Anda](#).
- Instal Docker di mesin lokal Anda. Anda memerlukan alat ini diinstal jika Anda ingin plugin membuat gambar kontainer untuk Anda dan mendorongnya ke repositori ECR. Atau Anda dapat melakukan tugas-tugas ini secara manual dan menginstruksikan plugin untuk menggunakan gambar kontainer yang ada. Untuk informasi selengkapnya tentang membuat gambar secara manual, lihat [Membuat gambar kontainer untuk Amazon GameLift Servers](#).

Untuk memulai alur kerja kontainer Amazon GameLift Servers terkelola:

- Di bilah alat utama editor Unity, pilih Amazon GameLift Servers menu, dan pilih Kontainer Terkelola. Tindakan ini membuka halaman plugin Host with Managed Containers, yang menyajikan step-by-step proses untuk membuat gambar kontainer dengan build server game Anda, menyebarkannya ke armada kontainer, dan meluncurkan game Anda.

Langkah 0: Atur profil Anda

Bagian ini menampilkan profil pengguna yang Anda pilih saat ini. Verifikasi bahwa profil pengguna saat ini adalah yang ingin Anda gunakan untuk alur kerja ini. Semua sumber daya yang Anda buat dalam alur kerja ini dikaitkan dengan AWS akun profil dan ditempatkan di AWS Wilayah default profil. Izin pengguna profil menentukan akses Anda ke AWS sumber daya dan tindakan.

Anda mungkin perlu mengubah profil pengguna yang dipilih jika:

- Tidak ada profil yang dipilih saat ini.

- Anda ingin memilih profil yang berbeda atau membuat profil baru.
- Anda perlu mem-bootstrap profil yang dipilih (jika status bootstrap tidak aktif).

Untuk mengatur atau mengubah profil pengguna yang dipilih

- Di Amazon GameLift Servers menu, pilih Open AWS Access Credentials.

Langkah 1: Nilai kesiapan wadah

Sebelum menerapkan server game Anda ke armada kontainer, Anda harus mengemasnya ke dalam gambar kontainer dan menyimpannya di repositori Amazon ECR. Plugin dapat menyelesaikan tugas-tugas ini untuk Anda atau Anda dapat melakukan tugas-tugas ini secara manual. Pada langkah ini, berikan informasi tentang status gambar kontainer Anda dan repositori ECR.

Gunakan pertanyaan penilaian untuk memberi tahu plugin langkah-langkah apa yang perlu diambil:

- Buat gambar kontainer baru. Jika Anda memilih opsi ini, langkah selanjutnya akan meminta Anda untuk lokasi direktori build server game Anda dan build yang dapat dieksekusi. Plugin menggunakan template Dockerfile (disediakan oleh Amazon GameLift Servers) dan secara otomatis mengkonfigurasinya untuk game Anda. Anda dapat melihat template di [Membangun gambar kontainer untuk Amazon GameLift Servers](#). Setelah memilih opsi ini, tunjukkan di mana Anda ingin plugin menyimpan gambar baru:
 - Buat repositori Amazon ECR baru dan dorong gambar kontainer ke sana. Plugin membuat repo ECR pribadi menggunakan AWS akun dan default Wilayah AWS di profil pengguna yang Anda pilih.
 - Dorong gambar kontainer ke repositori Amazon ECR yang dibuat sebelumnya. Jika Anda memilih opsi ini, langkah selanjutnya akan meminta Anda untuk memilih repositori Amazon ECR yang ada dari daftar. Daftar ini mencakup semua repositori Amazon ECR untuk AWS akun dan default Wilayah AWS di profil pengguna yang Anda pilih. Anda dapat memilih repositori publik atau pribadi.
- Gunakan gambar kontainer yang ada. Jika Anda telah membuat gambar secara manual, kami sarankan Anda menggunakan template Dockerfile yang disediakan oleh Amazon GameLift Servers, yang tersedia di [Membangun gambar kontainer untuk Amazon GameLift Servers](#). Setelah memilih opsi ini, tunjukkan di mana gambar berada.
 - Gambar buatan Docker yang disimpan secara lokal. Jika Anda memilih opsi ini, plugin membuat repositori pribadi Amazon ECR baru dan mendorong file gambar lokal ke sana. Langkah

selanjutnya akan meminta Anda untuk ID gambar, yang digunakan plugin untuk menemukan file gambar.

- Gambar kontainer yang sudah disimpan di repositori Amazon ECR. Jika Anda memilih opsi ini, langkah selanjutnya akan meminta Anda untuk memilih repositori Amazon ECR dan gambar yang ada dari daftar. Daftar ini mencakup semua repositori Amazon ECR untuk AWS akun dan default Wilayah AWS di profil pengguna yang Anda pilih. Anda dapat memilih repositori publik atau pribadi.

Langkah 2: Konfigurasi penyebaran gambar

Pada langkah ini, berikan informasi yang diperlukan plugin untuk menyebarkan gambar kontainer Anda ke armada kontainer. Langkah ini meminta informasi berikut:

- Lokasi pembuatan server game, image container, atau repositori Amazon ECR Anda, berdasarkan pilihan Anda di Langkah 1.
- Skenario yang akan digunakan untuk penerapan kontainer terkelola Anda.
- Pengaturan penyebaran opsional. Bagian ini memiliki pengaturan konfigurasi yang digunakan plugin secara default. Anda dapat memodifikasi ini atau menyimpan nilai default
 - Nama game diatur ke nama proyek game Anda secara default. Semua AWS sumber daya yang dibuat plugin mereferensikan nilai nama game.
 - Rentang port, batas memori, dan batas vCPU adalah pengaturan konfigurasi untuk armada kontainer. Untuk informasi selengkapnya tentang menyesuaikan nilai ini, lihat [Konfigurasi koneksi jaringan](#) rentang port koneksi, dan [Tetapkan batas sumber daya](#) untuk batas sumber daya.
 - Tag gambar kontainer digunakan untuk mengkategorikan gambar kontainer Anda di Amazon ECR. Nilai default-nya adalah `unity-gamelift-plugin`.

Opsi skenario penerapan

Armada kontainer satu wilayah

Skenario ini menyebarkan server game Anda ke armada kontainer tunggal. Ini adalah titik awal yang baik untuk menguji integrasi server Anda dengan AWS dan konfigurasi kontainer Anda. Ini menyebarkan sumber daya berikut.

- Amazon GameLift Servers definisi grup kontainer menjelaskan cara menerapkan dan menjalankan gambar kontainer Anda pada armada kontainer.

- Amazon GameLift Servers armada kontainer (On-Demand) dengan wadah server game Anda diinstal dan dijalankan, dengan alias.
- Kumpulan pengguna dan klien Amazon Cognito untuk memungkinkan pemain mengautentikasi dan memulai permainan.
- Authorizer API Gateway yang menautkan kumpulan pengguna dengan APIs.
- Daftar kontrol akses web (ACL) untuk membatasi panggilan pemain yang berlebihan ke API Gateway.
- Layanan backend untuk membuat permintaan ke Amazon GameLift Servers layanan atas nama klien game, seperti untuk meminta sesi permainan dan bergabung dengan game:
 - API Gateway + Fungsi Lambda bagi pemain untuk meminta slot sesi permainan. Fungsi ini memanggil `CreateGameSession()` jika tidak ada slot terbuka yang tersedia.
 - API Gateway + Fungsi Lambda bagi pemain untuk mendapatkan info koneksi untuk permintaan game mereka.

Armada kontainer satu wilayah dengan FlexMatch

Skenario ini menyebarkan server game Anda ke armada kontainer, mengonfigurasi penempatan sesi game, dan menyiapkan FlexMatch perjodohan. Skenario ini berguna ketika Anda siap untuk mulai merancang mak comblang khusus untuk solusi hosting Anda. Gunakan skenario ini untuk membuat sumber daya dasar untuk solusi ini, yang dapat Anda sesuaikan nanti sesuai kebutuhan. Ini menyebarkan sumber daya berikut:

- Amazon GameLift Servers definisi grup kontainer yang menjelaskan cara menerapkan dan menjalankan gambar kontainer Anda pada armada kontainer.
- Amazon GameLift Servers armada kontainer (On-Demand) dengan wadah server game Anda diinstal dan dijalankan, dengan alias.
- FlexMatch konfigurasi perjodohan dan aturan perjodohan ditetapkan untuk menerima permintaan pemain dan pertandingan formulir.
- Amazon GameLift Servers antrian sesi permainan yang memenuhi permintaan untuk pertandingan yang diusulkan dengan menemukan sumber daya hosting terbaik (berdasarkan kelayakan, biaya, latensi pemain, dll.) Dan memulai sesi permainan.
- Kumpulan pengguna dan klien Amazon Cognito untuk memungkinkan pemain mengautentikasi dan memulai permainan.
- Authorizer API Gateway yang menautkan kumpulan pengguna dengan APIs.

- Daftar kontrol akses web (ACL) untuk membatasi panggilan pemain yang berlebihan ke API Gateway.
- Layanan backend untuk membuat permintaan ke Amazon GameLift Servers layanan atas nama klien game, seperti untuk meminta sesi permainan dan bergabung dengan game:
 - API Gateway + Fungsi Lambda bagi pemain untuk meminta slot sesi permainan. Fungsi ini memanggil `StartMatchmaking()` jika tidak ada slot terbuka yang tersedia.
 - API Gateway + Fungsi Lambda bagi pemain untuk mendapatkan info koneksi untuk permintaan game mereka.
- Tabel DynamoDB untuk menyimpan tiket perjodohan untuk pemain dan informasi sesi permainan.
- Topik Amazon SNS+Lambda berfungsi untuk menangani acara. `GameSessionQueue`

Menyebarkan armada kontainer

Ketika konfigurasi armada Anda selesai, pilih tombol Deploy container fleet untuk memulai penyebaran. Proses ini dapat memakan waktu beberapa menit saat plugin membuat gambar kontainer dan mendorongnya ke ECR, menyediakan sumber daya hosting untuk armada kontainer, menyebarkan armada dan AWS sumber daya lainnya untuk skenario solusi hosting yang dipilih.

Saat memulai penerapan, Anda dapat melacak kemajuan setiap langkah. Tergantung pada konfigurasi Anda, langkah-langkahnya mungkin termasuk yang berikut:

- Mengkonfigurasi gambar kontainer
- Membuat repositori Amazon ECR
- Membangun gambar dan mendorong ke Amazon ECR
- Membuat definisi grup kontainer
- Membuat armada kontainer

Untuk informasi penerapan yang lebih detail, pilih Lihat di Konsol AWS Manajemen. Ketika armada kontainer mencapai status aktif, armada secara aktif menjalankan kontainer dengan proses server yang siap menjadi tuan rumah sesi permainan.

Saat penyebaran selesai, Anda memiliki armada kontainer yang berfungsi yang siap menyelenggarakan sesi permainan dan menerima koneksi pemain.

Anda tidak dapat menghentikan penerapan yang sedang berlangsung. Jika penerapan memasuki status buruk atau gagal, Anda dapat memulai kembali dengan menggunakan opsi Reset deployment.

Luncurkan klien

Pada titik ini, Anda telah menyelesaikan semua tugas untuk meluncurkan dan memainkan game multipemain Anda yang Amazon GameLift Servers dihosting. Untuk memainkan game Anda, pilih Mulai Klien untuk meluncurkan instance lokal klien game Anda.

- Jika Anda menerapkan skenario armada tunggal, buka satu instance klien game Anda dengan satu pemain dan masukkan peta server untuk bergerak. Anda dapat membuka instance kedua dari klien game untuk menambahkan pemain kedua ke peta game server yang sama.
- Jika Anda menerapkan FlexMatch skenario, solusi hosting menunggu setidaknya dua klien game untuk membuat permintaan perjodohan. Buka setidaknya dua contoh klien game Anda dengan satu pemain. Kedua pemain akan dicocokkan dan diminta untuk bergabung dengan sesi permainan untuk pertandingan.

Perbarui armada kontainer

Jika berhasil menerapkan solusi hosting kontainer terkelola, Anda dapat menggunakan fitur Update deployment. Opsi ini memungkinkan Anda memperbarui pengaturan konfigurasi untuk armada kontainer yang digunakan, tanpa harus membuat armada baru.

Saat memperbarui penerapan, Anda dapat menerapkan image kontainer dengan build server game yang berbeda, mengubah repositori Amazon ECR, memilih skenario penerapan yang berbeda, dan menyesuaikan pengaturan konfigurasi opsional.

Saat Anda siap untuk menerapkan perubahan, pilih Perbarui. Waktu yang diperlukan untuk pembaruan penerapan mirip dengan penerapan penuh. Untuk informasi penyebaran mendetail, pilih Lihat di Konsol AWS Manajemen.

Bersihkan sumber daya yang digunakan

Sebagai praktik terbaik, bersihkan AWS sumber daya untuk solusi kontainer terkelola Anda segera setelah Anda tidak lagi membutuhkannya. Anda mungkin terus mengeluarkan biaya untuk sumber daya ini jika Anda tidak menghapusnya.

Hapus sumber daya berikut:

- Tumpukan sumber daya kontainer terkelola. Sumber daya dalam tumpukan ini bergantung pada skenario penerapan yang Anda pilih. Untuk menghapus seluruh tumpukan, gunakan AWS CloudFormation konsol. Tumpukan yang dihasilkan dari Amazon GameLift Servers plugin menggunakan konvensi penamaan berikut: `GameLiftPluginForUnity-{GameName}-`

Containers. Tunggu proses penghapusan tumpukan selesai sebelum Anda memulai penerapan kontainer terkelola baru di plugin. Untuk informasi selengkapnya, lihat [Menghapus tumpukan dari CloudFormation konsol](#).

- **Repositori Amazon ECR.** Jika Anda menggunakan plugin untuk membuat repositori untuk gambar kontainer Anda, Anda mungkin ingin menghapus repositori yang tidak lagi diperlukan. Anda tidak perlu menghapus repositori sebelum mengatur ulang penerapan kontainer terkelola. Jika Anda memperbarui atau mengatur ulang penyebaran, plugin akan secara otomatis menggunakan repositori yang sama kecuali diarahkan untuk menggunakan yang lain. Untuk informasi selengkapnya, lihat [Menghapus repositori pribadi di Amazon ECR](#).

Amazon GameLift Serversplugin untuk Unity (server SDK 4.x)

Note

Topik ini memberikan informasi untuk versi Amazon GameLift Servers plugin yang lebih lama untuk Unity. Versi 1.0.0 (dirilis pada tahun 2021) menggunakan SDK server untuk Amazon GameLift Servers 4.x atau lebih lama. Untuk dokumentasi tentang versi terbaru plugin, yang menggunakan server SDK 5.x dan mendukung Amazon GameLift Servers Anywhere, lihat [Amazon GameLift Serversplugin untuk Unity \(server SDK 5.x\)](#)

Amazon GameLift Serversmenyediakan alat untuk mempersiapkan server game multipemain Anda untuk dijalankan. Amazon GameLift Servers Amazon GameLift ServersPlugin untuk Unity membuatnya lebih mudah untuk diintegrasikan Amazon GameLift Servers ke dalam proyek game Unity Anda dan menyebarkan Amazon GameLift Servers sumber daya untuk hosting cloud. Gunakan plugin Unity untuk mengakses Amazon GameLift Servers APIs dan menyebarkan AWS CloudFormation template untuk skenario game umum.

Setelah Anda mengatur plugin, Anda dapat mencoba [sampel Amazon GameLift Servers Unity](#) pada GitHub.

Topik

- [Integrasikan Amazon GameLift Servers dengan proyek server game Unity](#)
- [Integrasikan Amazon GameLift Servers dengan proyek klien game Unity](#)
- [Instal dan atur plugin](#)
- [Uji game Anda secara lokal](#)

- [Menyebarkan skenario](#)
- [Integrasikan game dengan Amazon GameLift Servers in Unity](#)
- [Impor dan jalankan contoh game](#)

Integrasikan Amazon GameLift Servers dengan proyek server game Unity

Note

Topik ini memberikan informasi untuk versi Amazon GameLift Servers plugin yang lebih lama untuk Unity. Versi 1.0.0 (dirilis pada tahun 2021) menggunakan SDK server untuk Amazon GameLift Servers 4.x atau lebih lama. Untuk dokumentasi tentang versi terbaru plugin, yang menggunakan server SDK 5.x dan mendukung Amazon GameLift Servers Anywhere, lihat [Amazon GameLift Servers plugin untuk Unity \(server SDK 5.x\)](#)

Topik ini membantu Anda mempersiapkan server game kustom Anda untuk hosting Amazon GameLift Servers. Server game harus dapat memberi tahu Amazon GameLift Servers tentang statusnya, untuk memulai dan menghentikan sesi permainan saat diminta, dan untuk melakukan tugas-tugas lain. Untuk informasi selengkapnya, lihat [Tambahkan Amazon GameLift Servers ke server game Anda](#).

Prasyarat

Sebelum mengintegrasikan server game Anda, selesaikan tugas-tugas berikut:

- [Menyiapkan peran layanan IAM untuk Amazon GameLift Servers](#)
- [Instal dan atur plugin](#)

Menyiapkan proses server baru

Note

Topik ini mengacu pada Amazon GameLift Servers plugin untuk Unity versi 1.0.0, yang menggunakan server SDK 4.x atau sebelumnya.

Siapkan komunikasi dengan Amazon GameLift Servers dan laporkan bahwa proses server siap untuk menjadi tuan rumah sesi permainan.

1. Inisialisasi SDK server dengan menelepon. `InitSDK()`
2. Untuk mempersiapkan server menerima sesi permainan, hubungi `ProcessReady()` dengan port koneksi dan detail lokasi sesi permainan. Sertakan nama fungsi callback yang dipanggil Amazon GameLift Servers layanan, seperti, `OnGameSession()`, `OnGameSessionUpdate()`, `OnProcessTerminate()`, `OnHealthCheck()` Amazon GameLift Servers mungkin perlu beberapa menit untuk memberikan panggilan balik.
3. Amazon GameLift Servers memperbarui status proses server ke `ACTIVE`.
4. Amazon GameLift Servers menelepon `onHealthCheck` secara berkala.

Contoh kode berikut menunjukkan cara mengatur proses server sederhana dengan 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;
}
```

Mulai sesi game

Note

Topik ini mengacu pada Amazon GameLift Servers plugin untuk Unity versi 1.0.0, yang menggunakan server SDK 4.x atau sebelumnya.

Setelah inisialisasi game selesai, Anda dapat memulai sesi permainan.

1. Menerapkan fungsi `onStartGameSession` callback. Amazon GameLift Servers memanggil metode ini untuk memulai sesi permainan baru pada proses server dan menerima koneksi pemain.
2. Untuk mengaktifkan sesi permainan, hubungi `ActivateGameSession()`. Untuk informasi selengkapnya tentang SDK, lihat [SDK server C # untuk Amazon GameLift Servers 4.x -- Tindakan](#).

Contoh kode berikut menggambarkan bagaimana memulai sesi permainan dengan 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();
}
```

Mengakhiri sesi game

Note

Topik ini mengacu pada Amazon GameLift Servers plugin untuk Unity versi 1.0.0, yang menggunakan server SDK 4.x atau sebelumnya.

Beri tahu Amazon GameLift Servers saat sesi permainan berakhir. Sebagai praktik terbaik, matikan proses server setelah sesi game selesai untuk mendaur ulang dan menyegarkan sumber daya hosting.

1. Siapkan fungsi bernama `onProcessTerminate` untuk menerima permintaan dari Amazon GameLift Servers dan panggilan `ProcessEnding()`.
2. Status proses berubah menjadi `TERMINATED`.

Contoh berikut menjelaskan cara mengakhiri proses untuk sesi permainan.

```
var processEndingOutcome = GameLiftServerAPI.ProcessEnding();

if (processReadyOutcome.Success)
    Environment.Exit(0);

// otherwise, exit with error code
Environment.Exit(errorCode);
```

Buat pembuatan server dan unggah ke Amazon GameLift Servers

Note

Topik ini mengacu pada Amazon GameLift Servers plugin untuk Unity versi 1.0.0, yang menggunakan server SDK 4.x atau sebelumnya.

Setelah Anda mengintegrasikan server game Anda Amazon GameLift Servers, unggah file build ke armada sehingga Amazon GameLift Servers dapat menerapkannya untuk hosting game. Untuk informasi selengkapnya tentang cara mengunggah server Anda Amazon GameLift Servers, lihat [Menerapkan build server khusus untuk Amazon GameLift Servers mengadakan](#).

Integrasikan Amazon GameLift Servers dengan proyek klien game Unity

Note

Topik ini memberikan informasi untuk versi Amazon GameLift Servers plugin yang lebih lama untuk Unity. Versi 1.0.0 (dirilis pada tahun 2021) menggunakan SDK server untuk Amazon GameLift Servers 4.x atau lebih lama. Untuk dokumentasi tentang versi terbaru plugin, yang menggunakan server SDK 5.x dan mendukung Amazon GameLift Servers Anywhere, lihat [Amazon GameLift Servers plugin untuk Unity \(server SDK 5.x\)](#)

Topik ini membantu Anda mengatur klien game untuk terhubung ke sesi game yang Amazon GameLift Servers dihosting melalui layanan backend. Gunakan Amazon GameLift Servers APIs untuk memulai perjodohan, meminta penempatan sesi game, dan banyak lagi.

Tambahkan kode ke proyek layanan backend untuk memungkinkan komunikasi dengan layanan. Amazon GameLift Servers Layanan backend menangani semua komunikasi klien game dengan layanan. GameLift Untuk informasi selengkapnya tentang layanan backend, lihat.

Server backend menangani tugas klien game berikut:

- Sesuaikan otentikasi untuk pemain Anda.
- Minta informasi tentang sesi permainan aktif dari Amazon GameLift Servers layanan.
- Buat sesi permainan baru.
- Tambahkan pemain ke sesi permainan yang ada.
- Hapus pemain dari sesi permainan yang ada.

Topik

- [Prasyarat](#)
- [Inisialisasi klien game](#)
- [Buat sesi permainan pada armada tertentu](#)
- [Tambahkan pemain ke sesi permainan](#)

- [Menghapus pemain dari sesi permainan](#)

Prasyarat

Sebelum mengatur komunikasi server game dengan Amazon GameLift Servers klien, selesaikan tugas-tugas berikut:

- [Mengatur sebuah Akun AWS](#)
- [Instal dan atur plugin](#)
- [Integrasikan Amazon GameLift Servers dengan proyek server game Unity](#)
- [Menyiapkan armada hosting dengan Amazon GameLift Servers](#)

Inisialisasi klien game

Note

Topik ini mengacu pada Amazon GameLift Servers plugin untuk Unity versi 1.0.0, yang menggunakan server SDK 4.x atau sebelumnya.

Tambahkan kode untuk menginisialisasi klien game. Jalankan kode ini saat diluncurkan, ini diperlukan untuk Amazon GameLift Servers fungsi lainnya.

1. Inisialisasi `AmazonGameLiftClient`. Panggil `AmazonGameLiftClient` dengan konfigurasi klien default atau konfigurasi khusus. Untuk informasi selengkapnya tentang cara mengkonfigurasi klien, lihat [Siapkan Amazon GameLift Servers pada layanan backend](#).
2. Hasilkan id pemain unik untuk setiap pemain untuk terhubung ke sesi permainan. Untuk mengetahui informasi selengkapnya, lihat [Hasilkan pemain IDs](#).

Contoh berikut menunjukkan cara mengatur Amazon GameLift Servers klien.

```
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);
}
}
```

Buat sesi permainan pada armada tertentu

 Note

Topik ini mengacu pada Amazon GameLift Servers plugin untuk Unity versi 1.0.0, yang menggunakan server SDK 4.x atau sebelumnya.

Tambahkan kode untuk memulai sesi game baru di armada yang Anda gunakan dan sediakan untuk pemain. Amazon GameLift Servers Setelah membuat sesi permainan baru dan mengembalikan `aGameSession`, Anda dapat menambahkan pemain ke dalamnya.

- Tempatkan permintaan untuk sesi permainan baru.

- Jika game Anda menggunakan armada, hubungi `CreateGameSession()` dengan ID armada atau alias, nama sesi, dan jumlah maksimum pemain bersamaan untuk game tersebut.
- Jika game Anda menggunakan antrian, hubungi `StartGameSessionPlacement()`

Contoh berikut menunjukkan cara membuat sesi permainan.

```
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;
}
```

Tambahkan pemain ke sesi permainan

Note

Topik ini mengacu pada Amazon GameLift Servers plugin untuk Unity versi 1.0.0, yang menggunakan server SDK 4.x atau sebelumnya.

Amazon GameLift Servers Setelah membuat sesi permainan baru dan mengembalikan `GameSession` objek, Anda dapat menambahkan pemain ke dalamnya.

1. Pesan slot pemain dalam sesi permainan dengan membuat sesi pemain baru. Gunakan `CreatePlayerSession` atau `CreatePlayerSessions` dengan ID sesi permainan dan ID unik untuk setiap pemain.
2. Connect ke sesi permainan. Ambil `PlayerSession` objek untuk mendapatkan informasi koneksi sesi permainan. Anda dapat menggunakan informasi ini untuk membuat koneksi langsung ke proses server:
 - a. Gunakan port yang ditentukan dan nama DNS atau alamat IP dari proses server.
 - b. Gunakan nama DNS dan port armada Anda. Nama dan port DNS diperlukan jika armada Anda mengaktifkan pembuatan sertifikat TLS.
 - c. Referensi ID sesi pemain. ID sesi pemain diperlukan jika server game Anda memvalidasi koneksi pemain yang masuk.

Contoh berikut menunjukkan cara memesan tempat pemain dalam sesi permainan.

```
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;
}
```

Kode berikut menggambarkan cara menghubungkan pemain dengan sesi permainan.

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

Menghapus pemain dari sesi permainan

Note

Topik ini mengacu pada Amazon GameLift Servers plugin untuk Unity versi 1.0.0, yang menggunakan server SDK 4.x atau sebelumnya.

Anda dapat menghapus pemain dari sesi permainan ketika mereka meninggalkan permainan.

1. Beri tahu Amazon GameLift Servers layanan bahwa pemain telah terputus dari proses server. Panggil `RemovePlayerSession` dengan ID sesi pemain.
2. Verifikasi `RemovePlayerSession` pengembalian `itsSuccess`. Kemudian, Amazon GameLift Servers ubah slot pemain agar tersedia, yang Amazon GameLift Servers dapat menetapkan ke pemain baru.

Contoh berikut menggambarkan cara menghapus sesi pemain.

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


Instal dan atur plugin

Note

Topik ini mengacu pada Amazon GameLift Servers plugin untuk Unity versi 1.0.0, yang menggunakan server SDK 4.x atau yang lebih lama.

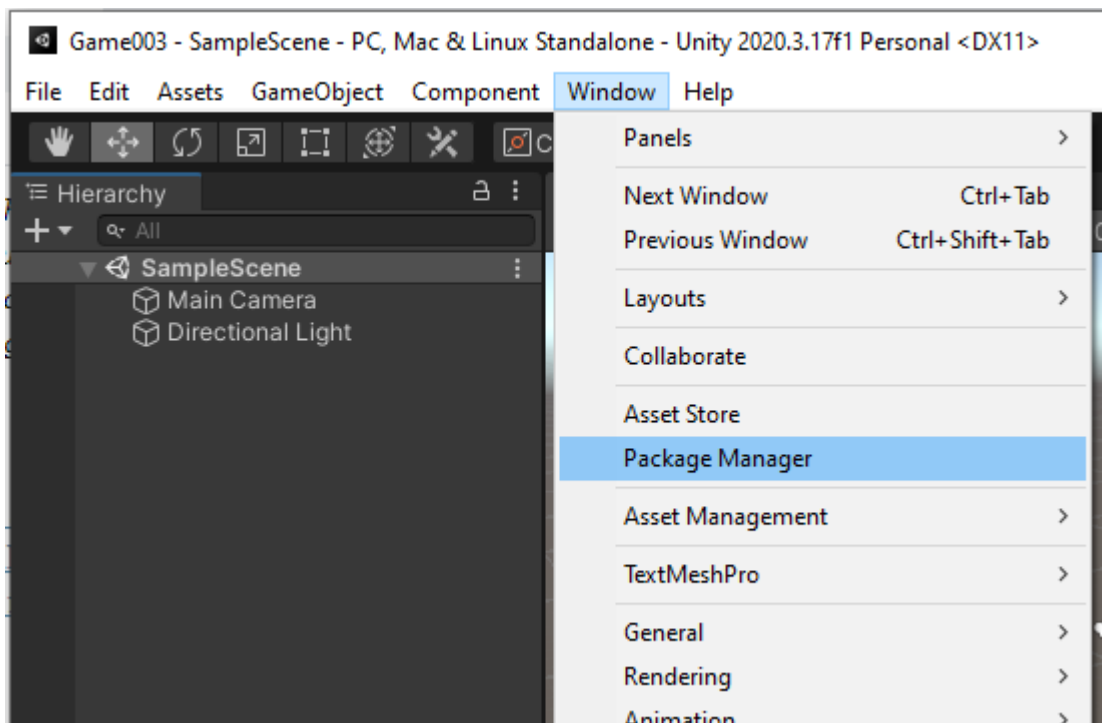
Bagian ini menjelaskan cara mengunduh, menginstal, dan mengatur Amazon GameLift Servers plugin untuk Unity, versi 1.0.0.

Prasyarat

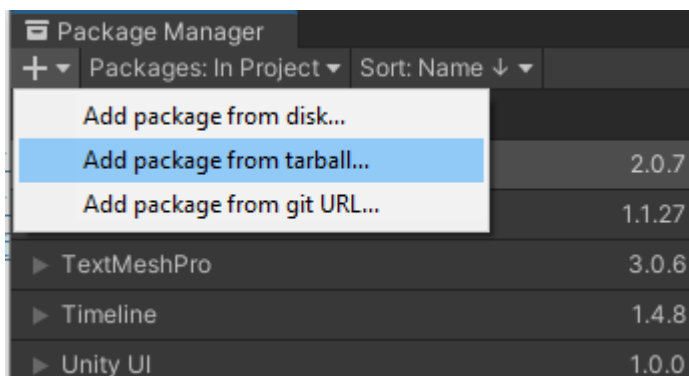
- Unity untuk Windows 2019.4 LTS, Windows 2020.3 LTS, atau Unity untuk macOS
- Versi Java saat ini
- Versi terkini dari .NET 4.x

Untuk mengunduh dan menginstal plugin untuk Unity

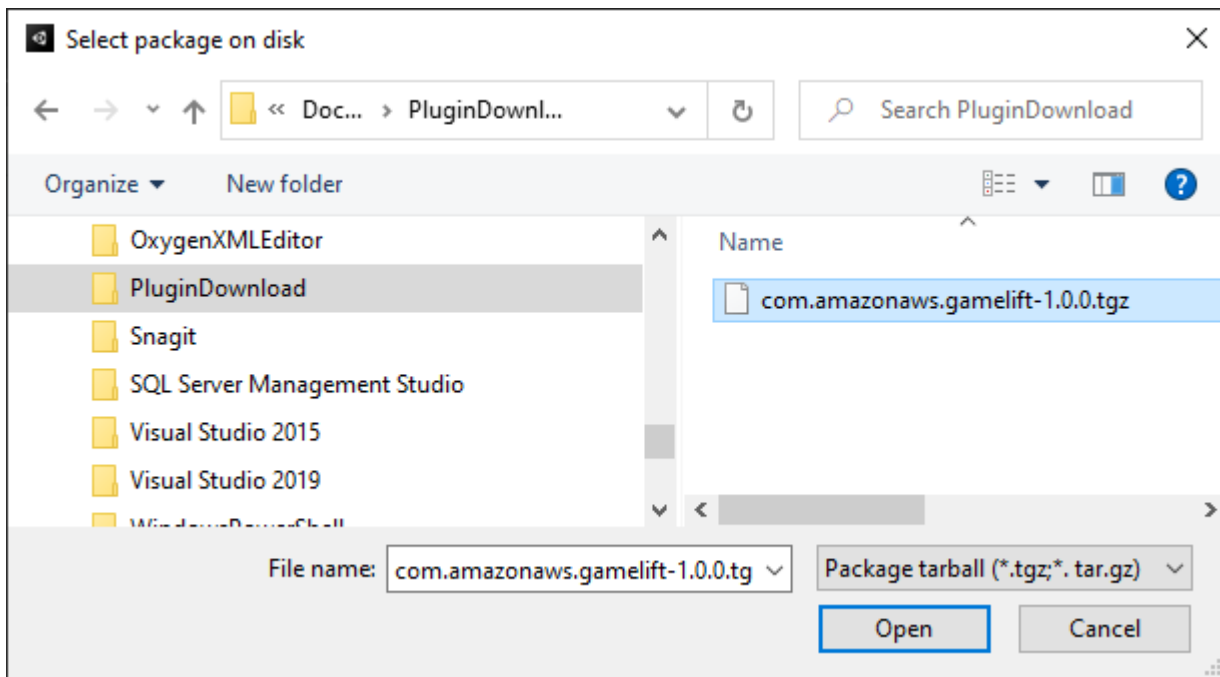
1. Unduh Amazon GameLift Servers plugin untuk Unity. Anda dapat menemukan versi terbaru di [Amazon GameLift Servers plugin untuk halaman repositori Unity](#). Di bawah [rilis terbaru](#), pilih Aset, lalu unduh `com.amazonaws.gamelift-version.tgz` file.
2. Luncurkan Unity dan pilih proyek.
3. Di bilah navigasi atas, di bawah Window pilih Package Manager:



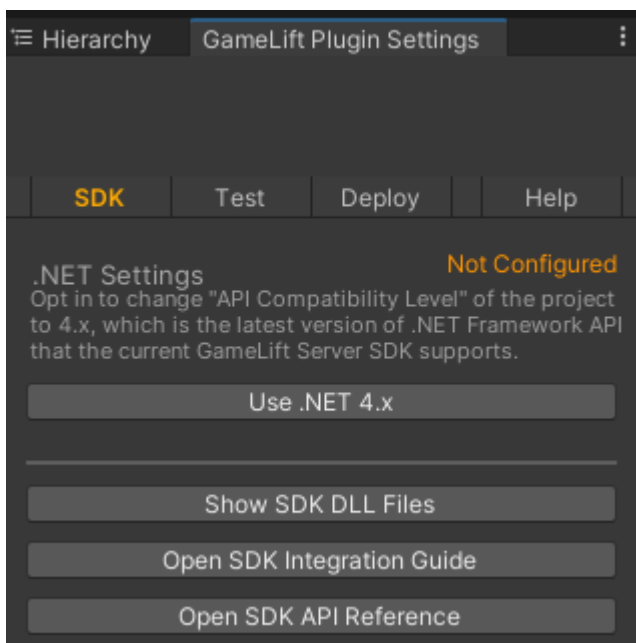
4. Di bawah tab Package Manager pilih +, lalu pilih Tambah paket dari tarball... :



5. Di jendela Pilih paket pada disk, arahkan ke `com.amazonaws.gamelift` folder, pilih `filecom.amazonaws.gamelift-version.tgz` , lalu pilih Buka:



6. Setelah Unity memuat plug-in, Amazon GameLift Servers muncul sebagai item baru di menu Unity. Mungkin perlu beberapa menit untuk menginstal dan mengkompilasi ulang skrip. Tab Pengaturan Amazon GameLift Servers Plugin secara otomatis terbuka.



7. Di panel SDK, pilih Use .NET 4.x.

Saat dikonfigurasi, status berubah dari Tidak Dikonfigurasi ke Dikonfigurasi.

Uji game Anda secara lokal

Note

Topik ini mengacu pada Amazon GameLift Servers plugin untuk Unity versi 1.0.0, yang menggunakan server SDK 4.x atau yang lebih lama.

Gunakan Amazon GameLift Servers Lokal untuk berjalan Amazon GameLift Servers di perangkat lokal Anda. Anda dapat menggunakan Amazon GameLift Servers Local untuk memverifikasi perubahan kode dalam hitungan detik, tanpa koneksi jaringan.

Konfigurasi pengujian lokal

1. Di jendela plugin untuk Unity, pilih tab Uji.
2. Di panel Uji, pilih Unduh Amazon GameLift Servers Lokal. Plugin untuk Unity membuka jendela browser dan mengunduh `GameLift_06_03_2021.zip` file ke folder unduhan Anda.

Unduhan mencakup C# Server SDK, file sumber.NET, dan komponen.NET yang kompatibel dengan Unity.

3. Unzip file `GameLift_06_03_2021.zip` yang diunduh.
4. Di jendela Pengaturan Amazon GameLift Servers Plugin, pilih Jalur Amazon GameLift Servers Lokal, arahkan ke folder yang tidak di-zip, pilih file **`GameLiftLocal.jar`**, lalu pilih Buka.

Saat dikonfigurasi, status pengujian lokal berubah dari Tidak Dikonfigurasi menjadi Dikonfigurasi.

5. Verifikasi status JRE. Jika statusnya Tidak Dikonfigurasi, pilih Unduh JRE dan instal versi Java yang direkomendasikan.

Setelah Anda menginstal dan mengkonfigurasi lingkungan Java, status berubah menjadi Dikonfigurasi.

Jalankan game lokal Anda

1. Di plugin untuk tab Unity, pilih tab Uji.
2. Di panel Uji, pilih Buka UI Uji Lokal.
3. Di jendela Pengujian Lokal, tentukan jalur yang dapat dieksekusi Server. Pilih... untuk memilih jalur dan nama yang dapat dieksekusi dari aplikasi server Anda.

4. Di jendela Pengujian Lokal, tentukan port GL Local.
5. Pilih Deploy & Run untuk menyebarkan dan menjalankan server.
6. Untuk menghentikan server game Anda, pilih Stop atau tutup jendela server game.

Menyebarkan skenario

Note

Topik ini mengacu pada Amazon GameLift Servers plugin untuk Unity versi 1.0.0, yang menggunakan server SDK 4.x atau yang lebih lama.

Skenario menggunakan AWS CloudFormation template untuk membuat sumber daya yang Anda butuhkan untuk menerapkan solusi cloud hosting untuk game Anda. Bagian ini menjelaskan skenario yang Amazon GameLift Servers disediakan dan cara menggunakannya.

Prasyarat

Untuk menerapkan skenario, Anda memerlukan peran IAM untuk layanan. Amazon GameLift Servers Untuk informasi tentang cara membuat peran Amazon GameLift Servers, lihat [Mengatur sebuah Akun AWS](#).

Setiap skenario memerlukan izin untuk sumber daya berikut:

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

Skenario

Note

Topik ini mengacu pada Amazon GameLift Servers plugin untuk Unity versi 1.0.0, yang menggunakan server SDK 4.x atau yang lebih lama.

Amazon GameLift Servers Plug-in untuk Unity mencakup skenario berikut:

Hanya autentikasi

Skenario ini menciptakan layanan backend game yang melakukan otentikasi pemain tanpa kemampuan server game. Template membuat sumber daya berikut di akun Anda:

- Kumpulan pengguna Amazon Cognito untuk menyimpan informasi otentikasi pemain.
- AWS Lambda Handler yang didukung titik akhir Amazon API Gateway REST yang memulai game dan melihat informasi koneksi game.

Armada Wilayah Tunggal

Skenario ini menciptakan layanan backend game dengan satu Amazon GameLift Servers armada. Ini menciptakan sumber daya berikut:

- Kumpulan pengguna Amazon Cognito bagi pemain untuk mengautentikasi dan memulai permainan.
- AWS Lambda Pawang untuk mencari sesi permainan yang ada dengan slot pemain terbuka di armada. Jika tidak dapat menemukan slot terbuka, itu menciptakan sesi permainan baru.

Armada Multi-Region dengan antrian dan mak comblang khusus

Skenario ini membentuk pertandingan dengan menggunakan Amazon GameLift Servers antrian dan mak comblang khusus untuk mengelompokkan pemain tertua di kolam tunggu. Ini menciptakan sumber daya berikut:

- Topik Layanan Pemberitahuan Sederhana Amazon yang Amazon GameLift Servers menerbitkan pesan ke. Untuk informasi selengkapnya tentang topik dan notifikasi SNS, lihat [Atur notifikasi kejadian untuk penempatan sesi game](#).

- Fungsi Lambda yang dipanggil oleh pesan yang mengkomunikasikan detail penempatan dan koneksi game.
- Tabel Amazon DynamoDB untuk menyimpan detail penempatan dan koneksi game. `GetGameConnection` panggilan dibaca dari tabel ini dan mengembalikan informasi koneksi ke klien game.

Temukan armada dengan antrian dan mak comblang khusus

Skenario ini membentuk kecocokan dengan menggunakan Amazon GameLift Servers antrian dan mak comblang khusus dan mengkonfigurasi tiga armada. Ini menciptakan sumber daya berikut:

- Dua armada Spot yang berisi jenis instans berbeda untuk memberikan daya tahan bagi ketidaktersediaan Spot.
- Armada On-Demand yang bertindak sebagai cadangan untuk armada Spot lainnya. Untuk informasi lebih lanjut tentang mendesain armada Anda, lihat [Sesuaikan Amazon GameLift Servers EC2 armada yang dikelola](#).
- Amazon GameLift Servers Antrian untuk menjaga ketersediaan server tetap tinggi dan biaya rendah. Untuk informasi selengkapnya dan praktik terbaik tentang antrian, lihat [Kustomisasi antrian sesi permainan](#)

FlexMatch

Skenario ini menggunakan FlexMatch, layanan perjudian terkelola, untuk mencocokkan pemain game bersama. Untuk informasi lebih lanjut tentang FlexMatch, lihat [Apa itu Amazon GameLift Servers FlexMatch](#). Skenario ini menciptakan sumber daya berikut:

- Fungsi Lambda untuk membuat tiket perjudian setelah menerima permintaan. `StartGame`
- Fungsi Lambda terpisah untuk mendengarkan acara FlexMatch pertandingan.

Untuk menghindari biaya yang tidak perlu pada Akun AWS, hapus sumber daya yang dibuat oleh setiap skenario setelah Anda selesai menggunakannya. Hapus AWS CloudFormation tumpukan yang sesuai.

Perbarui AWS kredensi

Note

Topik ini mengacu pada Amazon GameLift Servers plugin untuk Unity versi 1.0.0, yang menggunakan server SDK 4.x atau yang lebih lama.

Amazon GameLift ServersPlugin untuk Unity memerlukan kredensi keamanan untuk menerapkan skenario. Anda dapat membuat kredensial baru atau menggunakan kredensial yang ada.

Untuk informasi selengkapnya tentang mengonfigurasi kredensial, lihat [Memahami dan mendapatkan kredensial Anda. AWS](#)

Untuk memperbarui AWS kredensi

1. Di Unity, di plugin untuk tab Unity, pilih tab Deploy.
2. Di panel Deploy, pilih AWS Credentials.
3. Anda dapat membuat kredensial baru atau memilih AWS kredensial yang ada.
 - Untuk membuat kredensial, pilih Buat profil kredensial baru, lalu tentukan Nama Profil Baru, ID Kunci AWS Akses, Kunci AWS Rahasia, dan Wilayah AWS
 - Untuk memilih kredensi yang ada, pilih Pilih profil kredensial yang ada, lalu pilih nama profil dan Wilayah AWS
4. Di jendela Update AWS Credentials, pilih Update Credentials Profile.

Perbarui akun bootstrap

Note

Topik ini mengacu pada Amazon GameLift Servers plugin untuk Unity versi 1.0.0, yang menggunakan server SDK 4.x atau yang lebih lama.

Lokasi bootstrap adalah bucket Amazon S3 yang digunakan selama penerapan. Ini digunakan untuk menyimpan aset server game dan dependensi lainnya. Yang Wilayah AWS Anda pilih untuk bucket harus Region yang sama dengan yang akan Anda gunakan untuk penerapan skenario.

Untuk informasi selengkapnya tentang bucket Amazon S3, lihat [Membuat, mengonfigurasi, dan bekerja dengan bucket Amazon Simple Storage Service](#).

Untuk memperbarui lokasi akun bootstrap

1. Di Unity, di plugin untuk tab Unity, pilih tab Deploy.
2. Di panel Deploy, pilih Perbarui Bootstrap Akun.
3. Di jendela Bootstrapping Akun, Anda memilih bucket Amazon S3 yang ada atau membuat bucket Amazon S3 baru:
 - Untuk memilih bucket yang ada, pilih Pilih bucket Amazon S3 yang ada dan Perbarui untuk menyimpan pilihan Anda.
 - Pilih Buat bucket Amazon S3 baru untuk membuat bucket Amazon Simple Storage Service baru, lalu pilih Kebijakan. Kebijakan menentukan kapan bucket Amazon S3 akan kedaluwarsa. Pilih Buat untuk membuat ember.

Menyebarkan skenario permainan

 Note

Topik ini mengacu pada Amazon GameLift Servers plugin untuk Unity versi 1.0.0, yang menggunakan server SDK 4.x atau yang lebih lama.

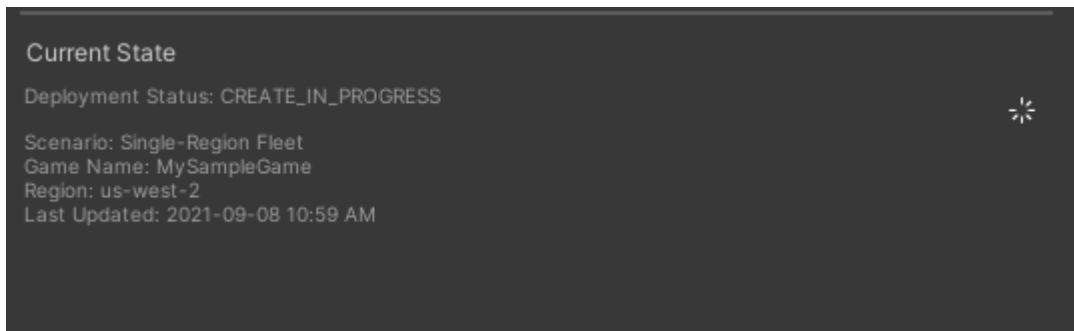
Anda dapat menggunakan skenario untuk menguji permainan AndaAmazon GameLift Servers. Setiap skenario menggunakan AWS CloudFormation template untuk membuat tumpukan dengan sumber daya yang diperlukan. Sebagian besar skenario memerlukan server game yang dapat dieksekusi dan membangun jalur. Saat Anda menerapkan skenario, Amazon GameLift Servers menyalin aset game ke lokasi bootstrap sebagai bagian dari penerapan.

Anda harus mengonfigurasi AWS kredensi dan bootstrap AWS akun untuk menerapkan skenario.

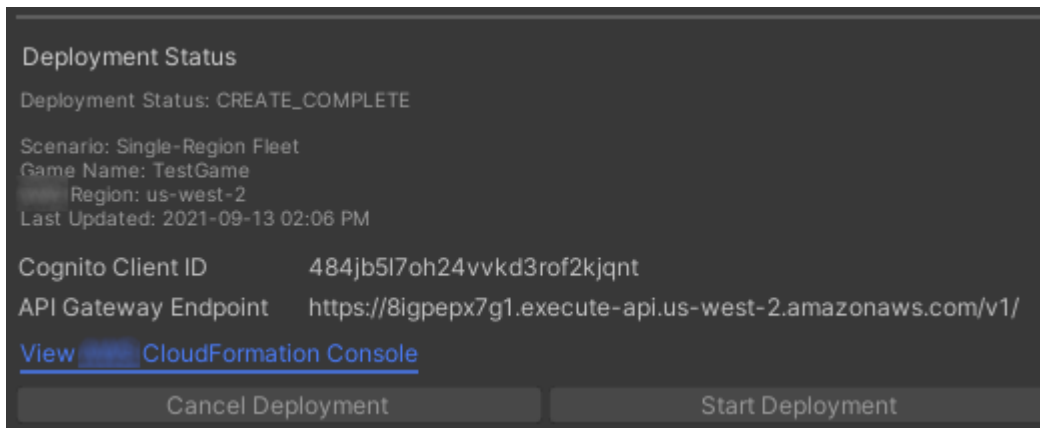
Untuk menyebarkan skenario

1. Di Unity, di plugin untuk tab Unity, pilih tab Deploy.
2. Di panel Deploy, pilih Open Deployment UI.
3. Di jendela Deployment, pilih skenario.

4. Masukkan Nama Game. Nama ini harus unik. Nama game adalah bagian dari nama AWS CloudFormation tumpukan saat Anda menerapkan skenario.
5. Pilih Jalur Folder Build Server Game. Jalur folder build menunjuk ke folder yang berisi server yang dapat dieksekusi dan dependensi.
6. Pilih Jalur File Bangun.exe Server Game. Jalur file yang dapat dieksekusi build menunjuk ke server game yang dapat dieksekusi.
7. Pilih Mulai Deployment untuk mulai menerapkan skenario. Anda dapat mengikuti status pembaruan di jendela Deployment di bawah Keadaan Saat Ini. Skenario dapat memakan waktu beberapa menit untuk digunakan.



8. Saat skenario menyelesaikan penerapan, Status Saat Ini akan diperbarui untuk menyertakan ID Klien Cognito dan Titik Akhir Gateway API yang dapat Anda salin dan tempel ke dalam game.



9. Untuk memperbarui pengaturan game, pada menu Unity, pilih Buka Pengaturan Koneksi Klien. Ini menampilkan tab Inspector di sisi kanan layar Unity.
10. Hapus pilihan Mode Pengujian Lokal.
11. Masukkan Endpoint API Gateway dan ID Klien Cognito. Pilih yang sama dengan yang Wilayah AWS Anda gunakan untuk penerapan skenario. Anda kemudian dapat membangun kembali dan menjalankan klien game menggunakan sumber daya skenario yang diterapkan.

Menghapus sumber daya yang dibuat oleh skenario

Note

Topik ini mengacu pada Amazon GameLift Servers plugin untuk Unity versi 1.0.0, yang menggunakan server SDK 4.x atau yang lebih lama.

Untuk menghapus sumber daya yang dibuat untuk skenario, hapus AWS CloudFormation tumpukan yang sesuai.

Untuk menghapus sumber daya yang dibuat oleh skenario

1. Di jendela Amazon GameLift Servers plugin untuk Unity Deployment, pilih View AWS CloudFormation Console untuk membuka AWS CloudFormation konsol.
2. Di AWS CloudFormation konsol, pilih Tumpukan, lalu pilih tumpukan yang menyertakan nama game yang ditentukan selama penerapan.
3. Pilih Hapus untuk menghapus tumpukan. Mungkin perlu beberapa menit untuk menghapus tumpukan. Setelah AWS CloudFormation menghapus tumpukan yang digunakan oleh skenario, statusnya berubah menjadi ROLLBACK_COMPLETE.

Integrasikan game dengan Amazon GameLift Servers in Unity

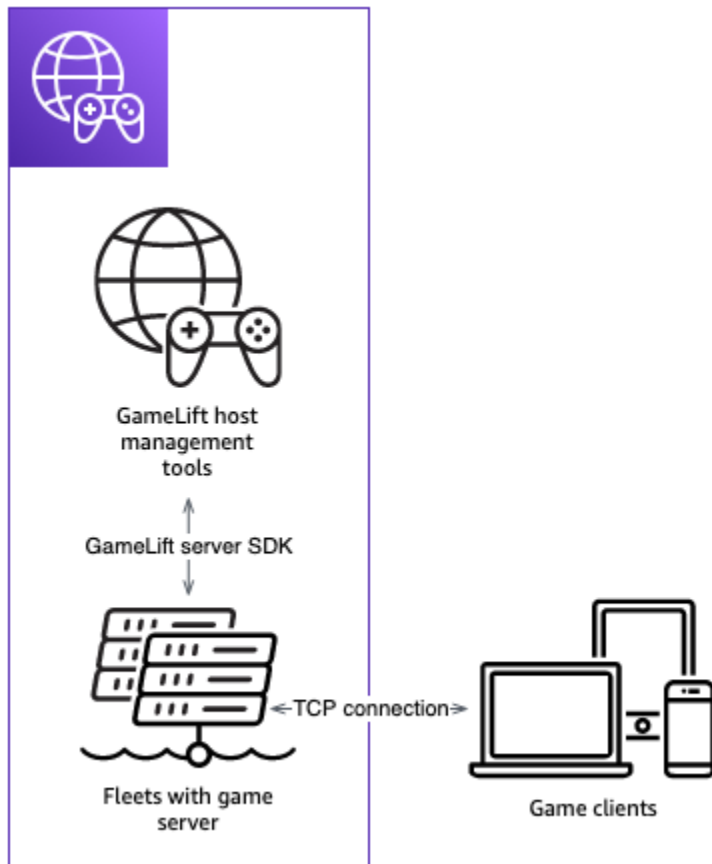
Note

Topik ini mengacu pada Amazon GameLift Servers plugin untuk Unity versi 1.0.0, yang menggunakan server SDK 4.x atau yang lebih lama.

Integrasikan game Unity Anda Amazon GameLift Servers dengan menyelesaikan tugas-tugas berikut:

- [Integrasikan Amazon GameLift Servers dengan proyek server game Unity](#)
- [Integrasikan Amazon GameLift Servers dengan proyek klien game Unity](#)

Diagram berikut menunjukkan aliran contoh mengintegrasikan permainan. Dalam diagram, armada dengan server game diarahkan ke Amazon GameLift Servers. Klien game berkomunikasi dengan server game, yang berkomunikasi dengannya. Amazon GameLift Servers



Impor dan jalankan contoh game

Note

Topik ini mengacu pada Amazon GameLift Servers plugin untuk Unity versi 1.0.0, yang menggunakan server SDK 4.x atau yang lebih lama.

Amazon GameLift ServersPlugin untuk Unity mencakup contoh permainan yang dapat Anda gunakan untuk menjelajahi dasar-dasar mengintegrasikan game Anda. Amazon GameLift Servers Di bagian ini, Anda membangun klien game dan server game dan kemudian menguji secara lokal menggunakan Amazon GameLift Servers Lokal.

Prasyarat

- [Mengatur sebuah Akun AWS](#)
- [Instal dan atur plugin](#)

Membangun dan menjalankan server game sampel

Note

Topik ini mengacu pada Amazon GameLift Servers plugin untuk Unity versi 1.0.0, yang menggunakan server SDK 4.x atau yang lebih lama.

Siapkan file server game dari game sampel.

1. Di Unity, pada menu, pilih Amazon GameLift Servers, lalu pilih Impor Contoh Game.
2. Di jendela Impor Contoh Game, pilih Impor untuk mengimpor game, aset, dan dependensinya.
3. Bangun server game. Di Unity, pada menu, pilih Amazon GameLift Servers, lalu pilih Apply Windows Sample Server Build Settings atau Apply macOS Sample Server Build Settings. Setelah Anda mengonfigurasi pengaturan server game, Unity mengompilasi ulang aset.
4. Di Unity, pada menu, pilih File, lalu pilih Build. Pilih Server Build, pilih Build, lalu pilih folder build khusus untuk file server.

Unity membangun server game sampel, menempatkan aset yang dapat dieksekusi dan diperlukan di folder build yang ditentukan.

Bangun dan jalankan klien game sampel

Note

Topik ini mengacu pada Amazon GameLift Servers plugin untuk Unity versi 1.0.0, yang menggunakan server SDK 4.x atau yang lebih lama.

Siapkan file klien game dari game sampel.

1. Di Unity, pada menu, pilih Amazon GameLift Servers, lalu pilih Apply Windows Sample Client Build Settings atau Apply macOS Sample Client Build Settings. Setelah pengaturan klien game dikonfigurasi, Unity akan mengompilasi ulang aset.
2. Di Unity, pada menu, pilih Go To Client Settings. Ini akan menampilkan tab Inspector di sisi kanan layar Unity. Di tab Pengaturan Amazon GameLift Servers Klien, pilih Mode Pengujian Lokal.

3. Bangun klien game. Di Unity, pada menu, pilih File. Konfirmasi Build Server tidak dicentang, pilih Build, lalu pilih folder build khusus untuk file klien.

Unity membangun klien game sampel, menempatkan aset yang dapat dieksekusi dan diperlukan di folder build klien yang ditentukan.

4. Anda tidak membangun server game dan klien. Pada langkah selanjutnya, Anda menjalankan game dan melihat bagaimana ia berinteraksi dengannya Amazon GameLift Servers.

Uji contoh permainan secara lokal

Note

Topik ini mengacu pada Amazon GameLift Servers plugin untuk Unity versi 1.0.0, yang menggunakan server SDK 4.x atau yang lebih lama.

Jalankan contoh permainan yang Anda impor menggunakan Amazon GameLift Servers Lokal.

1. Luncurkan server game. Di Unity, di plugin untuk tab Unity, pilih tab Deploy.
2. Di panel Uji, pilih Buka UI Uji Lokal.
3. Di jendela Pengujian Lokal, tentukan Game Server .exe File Path. Jalur harus menyertakan nama yang dapat dieksekusi. Misalnya, `C:/MyGame/GameServer/MyGameServer.exe`.
4. Pilih Deploy dan Run. Plugin untuk Unity meluncurkan server game dan membuka jendela log Amazon GameLift Servers Lokal. Jendela berisi pesan log termasuk pesan yang dikirim antara server game dan Amazon GameLift Servers Lokal.
5. Luncurkan klien game. Temukan lokasi pembuatan dengan klien game sampel dan pilih file yang dapat dieksekusi.
6. Di Game Amazon GameLift Servers Contoh, berikan email dan kata sandi lalu pilih Masuk. Email dan kata sandi tidak divalidasi atau digunakan.
7. Di Game Amazon GameLift Servers Contoh, pilih Mulai. Klien game mencari sesi permainan. Jika tidak dapat menemukan sesi, itu membuatnya. Klien game kemudian memulai sesi permainan. Anda dapat melihat aktivitas game di log.

Contoh log server game

...

```

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

```

Contoh log Amazon GameLift Servers lokal

```

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/
gssess-59f6cc44-4361-42f5-95b5-fdb5825c0f3d
12:56:24,266 INFO || - [WebSocketInvoker] Thread-12 - StartGameSessionRequest:
gameSessionId=arn:aws:gamelift:local::gamesession/fleet-123/
gssess-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/gssess-59f6cc44-4361-42f5-95b5-
fdb5825c0f3d created

```



```
12:56:24,585 INFO || - [GameLiftHttpHandler] Thread-12 - API to use:
GameLift.DescribeGameSessions
12:56:24,585 INFO || - [DescribeGameSessionsDispatcher] Thread-12 - Received API call
to describe game sessions with input: {"FleetId":"fleet-123"}
12:56:24,660 INFO || - [GameLiftSdkHttpHandler] GameLiftSdkHttpHandler-thread-0 -
GameLift API to use: GameSessionActivate for pId 20728
12:56:24,661 INFO || - [GameSessionActivateHandler] GameLiftSdkHttpHandler-thread-0 -
Received API call for GameSessionActivate from 20728
12:56:24,678 INFO || - [GameSessionActivateHandler] GameLiftSdkHttpHandler-thread-0
- GameSessionActivate data: gameId: "arn:aws:gamelift:local::gamesession/
fleet-123/gsess-59f6cc44-4361-42f5-95b5-fdb5825c0f3d"
```

Matikan proses server

Note

Topik ini mengacu pada Amazon GameLift Servers plugin untuk Unity versi 1.0.0, yang menggunakan server SDK 4.x atau yang lebih lama.

Setelah selesai dengan game sampel, matikan server di Unity.

1. Di klien game, pilih Keluar atau tutup jendela untuk menghentikan klien game.
2. Di Unity, di jendela Pengujian Lokal, pilih Berhenti atau tutup jendela server game untuk menghentikan server.

Peta jalan pengembangan untuk hosting dengan Amazon GameLift Servers terkelola EC2

Peta jalan ini memandu Anda melalui cara mengembangkan solusi EC2 hosting Amazon GameLift Servers terkelola untuk game multipemain Anda. Amazon GameLift Servers menawarkan beberapa opsi hosting game; untuk informasi lebih lanjut tentang opsi ini, lihat [Amazon GameLift Servers solusi](#).

Dengan hosting Amazon GameLift Servers terkelola, server game Anda di-host pada sumber daya komputasi virtual AWS Cloud berbasis yang Amazon GameLift Servers memiliki dan beroperasi berdasarkan konfigurasi Anda. Anda mendapatkan keamanan, keandalan, dan ketersediaan global instans Amazon Elastic Compute Cloud (Amazon EC2) yang dioptimalkan lebih lanjut untuk digunakan dengan hosting game multipemain. Amazon GameLift Servers merampingkan manajemen

hosting dengan alat-alat seperti penerapan server otomatis, penanganan siklus hidup, dan auto-scaling kapasitas.

Solusi Amazon GameLift Servers terkelola terdiri dari komponen-komponen berikut:

- Satu atau beberapa armada Amazon GameLift Servers terkelola, yang menggunakan instans Amazon Elastic Compute Cloud EC2 (Amazon) yang dioptimalkan untuk hosting game multipemain.
- Sebuah build server game, terintegrasi dengan SDK server untuk Amazon GameLift Servers, untuk digunakan di semua armada.
- Klien game dan layanan backend, terintegrasi dengan AWS SDK, untuk berinteraksi dengan Amazon GameLift Servers layanan dan meminta sesi permainan.
- Amazon GameLift Servers Antrian untuk menempatkan sesi permainan baru dengan server game yang tersedia di semua armada.
- (Opsional) FlexMatch Mak comblang untuk membuat pertandingan multi-pemain dan mengatur sesi permainan untuk mereka.

Peta jalan ini menyajikan jalur yang efisien untuk membuat game multipemain Anda aktif dan berjalan dengan sukses dengan Amazon GameLift Servers hosting terkelola. EC2 Setelah Anda memiliki komponen yang diperlukan, Anda dapat terus mengulangi pengembangan game dan menyesuaikan solusi hosting Anda. Saat Anda semakin dekat dengan peluncuran, lihat ini [Mempersiapkan game Anda untuk diluncurkan dengan Amazon GameLift Servers hosting](#) untuk bantuan dalam mempersiapkan solusi hosting Anda untuk penggunaan tingkat produksi.

i Mulailah dengan Amazon GameLift Servers plugin untuk Unreal Engine dan Unity

Untuk penyebaran yang lebih cepat, coba [Amazon GameLift Servers plugin](#) untuk Unreal Engine dan Unity. Ini menyediakan alur kerja UI terpandu untuk menyebarkan server game Anda dengan cepat dengan pengaturan minimal, sehingga Anda dapat mencoba komponen game Anda dalam aksi. Kemudian Anda dapat membangun fondasi ini untuk membuat solusi hosting khusus untuk game Anda. Untuk detail selengkapnya, lihat [Jelajahi dengan Amazon GameLift Servers plugin](#).

Langkah 1: Siapkan server game Anda untuk bekerja Amazon GameLift Servers

Tambahkan fungsionalitas ke server game Anda sehingga dapat berkomunikasi dengan Amazon GameLift Servers layanan saat digunakan untuk hosting.

- Dapatkan SDK server untuk Amazon GameLift Servers (versi 5.x) untuk proyek game Anda. Server SDK tersedia dalam C ++, C #, dan Go. [Unduh SDK Amazon GameLift Servers server](#).
- Ubah kode server game Anda untuk menambahkan fungsionalitas SDK server. Untuk panduan, lihat [Integrasikan game dengan server game khusus](#). Minimal, lakukan hal berikut:
 - Tambahkan kode untuk menginisialisasi Amazon GameLift Servers SDK dan membuat WebSocket koneksi dengan layanan. Amazon GameLift Servers Gunakan tindakan `InitSdk()` SDK server.
 - Tambahkan kode untuk melaporkan ke Amazon GameLift Servers layanan ketika proses server siap untuk meng-host sesi permainan. Gunakan tindakan `ProcessReady()` SDK server.
 - Menerapkan fungsi callback yang diperlukan `OnProcessTerminate()`, dan `OnStartGameSession()`. Dengan fungsi-fungsi ini, proses server game dapat mempertahankan koneksi dengan Amazon GameLift Servers layanan, memulai sesi permainan ketika diminta oleh Amazon GameLift Servers, dan menanggapi prompt untuk mengakhiri proses server game.
 - Tambahkan kode untuk melaporkan ke Amazon GameLift Servers layanan saat proses server mengakhiri sesi permainan. Gunakan tindakan `ProcessEnding()` SDK server.
- Package build server game Anda. Buat skrip penginstalan dengan file build, dependensi, dan perangkat lunak pendukung Anda. Lihat [Membuat paket file bangunan game Anda](#). Sebaiknya gunakan bucket Amazon Simple Storage Service (Amazon S3) untuk menyimpan versi build game Anda.
- Uji integrasi server game Anda. Untuk tugas ini, kami sarankan menyiapkan armada Amazon GameLift Servers Anywhere untuk workstation lokal, seperti yang dijelaskan dalam [Siapkan pengujian lokal dengan Amazon GameLift Servers Anywhere](#). Untuk langkah ini, instal server game Anda secara manual ke perangkat uji dan mulai proses server. Gunakan AWS CLI untuk meminta sesi permainan baru, dan verifikasi bahwa Amazon GameLift Servers layanan berhasil meminta proses server Anda untuk memulai sesi permainan.

Langkah 2: Siapkan klien game Anda untuk bergabung dengan sesi game yang dihosting

Buat cara bagi klien game Anda untuk meminta bergabung dengan sesi game, mendapatkan info koneksi, dan kemudian terhubung langsung ke sesi game yang dihosting. Pendekatan yang paling umum adalah mengatur fungsionalitas layanan backend yang berfungsi sebagai perantara antara klien game Anda dan layanan. Amazon GameLift Servers Pendekatan ini melindungi sumber daya hosting Anda dan memberi Anda kontrol yang lebih besar atas bagaimana pemain ditempatkan ke dalam sesi permainan.

- Bangun fungsionalitas layanan backend untuk hosting. Layanan backend berkomunikasi dengan Amazon GameLift Servers layanan dan memberikan informasi koneksi ke klien game. Fungsionalitas ini termasuk memulai sesi permainan, menempatkan pemain ke dalam game, dan mengambil informasi sesi permainan. Untuk panduan, lihat [Integrasikan game dengan server game khusus](#). Minimal, lakukan hal berikut:
 - Dapatkan AWS SDK untuk Amazon GameLift Servers dan tambahkan ke proyek layanan backend Anda. Lihat [sumber daya Amazon GameLift Servers SDK untuk layanan klien](#).
 - Tambahkan kode untuk menginisialisasi Amazon GameLift Servers klien dan menyimpan pengaturan kunci. Lihat [Siapkan Amazon GameLift Servers pada layanan backend](#).
 - Tambahkan fungsionalitas untuk memanggil aksi AWS SDK `CreateGameSession()` dan memberikan informasi koneksi sesi game ke klien game. Lihat [Membuat sesi permainan pada armada tertentu](#).

Memanggil `CreateGameSession()` adalah titik awal yang nyaman untuk meminta sesi permainan baru. Setelah Anda memiliki sistem penempatan sesi permainan di tempat (lihat Langkah 3), Anda akan mengganti kode ini dengan panggilan ke `StartGameSessionPlacement()` (atau `StartMatchmaking()` jika Anda menggunakan `FlexMatch`).

Untuk panduan tentang merancang layanan backend Anda, lihat. [Rancang layanan klien game Anda](#)

- Tambahkan fungsionalitas ke klien game Anda yang memungkinkan pemain bergabung dengan sesi game yang dihosting. Klien game membuat permintaan ke layanan backend Anda, tidak langsung ke Amazon GameLift Servers. Setelah layanan backend menyediakan informasi koneksi sesi game, klien game terhubung langsung dengan sesi game untuk memainkan game.
- Uji integrasi klien game Anda. Anda dapat menggunakan armada Amazon GameLift Servers Anywhere yang sama dengan workstation lokal untuk pengujian.

Untuk pengembangan berulang yang cepat atau saat bekerja dengan tim multi-orang, sebaiknya siapkan lingkungan pengujian [berbasis cloud](#). Solusi Amazon GameLift Servers Toolkit ini meniru perilaku armada Amazon GameLift Servers terkelola tetapi memungkinkan Anda memperbarui build server game dengan waktu penyelesaian minimal.

Langkah 3: Mengatur penempatan sesi permainan

Sesuaikan cara Anda Amazon GameLift Servers ingin memproses permintaan untuk sesi permainan baru dan temukan server game yang tersedia untuk meng-host mereka. Amazon GameLift Servers secara otomatis melacak ketersediaan semua server game di semua armada. Ketika klien game mengirim permintaan untuk bergabung dengan sesi permainan, Amazon GameLift Servers cari penempatan “terbaik” berdasarkan serangkaian prioritas yang ditentukan seperti latensi minimum, biaya, dan ketersediaan.

- Buat antrian sesi permainan untuk menempatkan sesi permainan baru dengan server game yang tersedia. Antrian adalah mekanisme utama untuk penempatan sesi permainan. Untuk panduan, lihat [Buat antrian sesi game](#).
- Minimal, tambahkan armada Anywhere Anda sebagai tujuan dalam antrian Anda. Semua pengaturan lainnya adalah kustomisasi opsional.
- Dalam kode layanan backend Anda, ubah **CreateGameSession()** panggilan ke **StartGameSessionPlacement()** Lihat [Membuat sesi permainan dalam antrian multi-lokasi](#).
- Buat mekanisme untuk memberi tahu klien game saat sesi permainan siap untuk bergabung. Saat dalam pengembangan, Anda dapat melakukan polling untuk status sesi game menggunakan panggilan ke DescribeGameSessionPlacement. Namun, sebelum menggunakan antrian untuk memproses volume tinggi, Anda harus mengaktifkan notifikasi acara. Lihat [Atur notifikasi kejadian untuk penempatan sesi game](#).
- (Opsional) Tambahkan komponen FlexMatch perjudian. Untuk panduan, lihat [panduan Amazon GameLift Servers FlexMatch pengembang](#).

Langkah 4: Buat armada berbasis cloud terkelola

Hingga saat ini Anda telah bekerja dengan armada Anywhere yang dikelola sendiri untuk menguji dan mengulangi komponen game Anda dan Anda telah menyempurnakan penempatan sesi game Anda. Bagian terakhir dari solusi Anda adalah mengatur jenis sumber daya hosting yang Anda

perlu untuk sistem produksi. Untuk mulai merencanakan dan mengonfigurasi produksi, Anda ingin beralih ke bekerja dengan armada yang Amazon GameLift Servers dikelola.

- Package build server game Anda dan unggah ke Amazon GameLift Servers. Buat skrip penginstalan dengan file build, dependensi, dan perangkat lunak pendukung Anda. Lihat [Menerapkan build server khusus untuk Amazon GameLift Servers mengadakan](#). Anda dapat mengunggah build Anda Amazon GameLift Servers menggunakan konsol atau AWS CLI.

Sebelum mengunggah build Anda, putuskan apa yang ingin Wilayah AWS Anda buat armada. Anda harus mengunggah build ke Wilayah yang sama. Untuk informasi lebih lanjut tentang memilih lokasi armada, lihat [Lokasi armada](#).

- Buat EC2 armada yang dikelola. Saat Anda membuat armada, Amazon GameLift Servers segera mulai menerapkan build server game Anda untuk hosting. Anda dapat mengonfigurasi banyak aspek armada yang dikelola. Untuk panduan, lihat [Buat EC2 armada Amazon GameLift Servers terkelola](#). Minimal, lakukan hal berikut:
 - Beri nama armada dan tentukan build game yang diunggah untuk diterapkan.
 - Pilih Instans Sesuai Permintaan untuk armada Anda dan pilih jenis instans yang tersedia di lokasi armada Anda. Armada spot adalah pilihan yang berharga tetapi membutuhkan desain dan konfigurasi tambahan.
 - Buat konfigurasi runtime untuk armada. Minimal, tentukan jalur peluncuran untuk server game Anda yang dapat dieksekusi.
 - Tentukan pengaturan port untuk memungkinkan lalu lintas masuk mengakses server game Anda.
- Tambahkan armada terkelola ke antrian Anda. Dalam antrian sesi permainan Anda, ganti armada Anywhere dengan armada terkelola Anda.
- Uji hosting game dengan armada terkelola Anda. Pada titik ini Anda harus dapat menguji seluruh siklus hosting, dengan klien game yang meminta sesi permainan, mendapatkan info koneksi, dan berhasil terhubung ke sesi permainan.

Langkah 5: Sesuaikan armada terkelola Anda

Saat Anda mempersiapkan peluncuran game, Anda harus menyempurnakan sumber daya hosting terkelola Anda. Beberapa keputusan yang perlu dipertimbangkan meliputi:

- Pertimbangkan untuk menambahkan armada Spot untuk penghematan biaya. Lihat [Tutorial: Membuat Amazon GameLift Servers antrian dengan Instans Spot](#).

- Jika server game Anda perlu mengkomunikasikan AWS sumber daya lain, siapkan peran IAM untuk mengelola akses. Lihat [Berkomunikasi dengan AWS sumber daya lain dari armada Anda](#).
- Tentukan di mana secara geografis Anda ingin memposisikan server game. Tambahkan lokasi jarak jauh ke armada terkelola Anda. Lihat [Sesuaikan Amazon GameLift Servers EC2 armada yang dikelola](#).
- Optimalkan kinerja armada dengan memilih jenis dan ukuran instans dan mengonfigurasi runtime untuk menjalankan beberapa proses server. Lihat [Kelola caranya Amazon GameLift Servers meluncurkan server game](#).
- Bereksperimenlah dengan opsi penempatan sesi game untuk armada terkelola, termasuk menyesuaikan pengaturan prioritas. Lihat [Kustomisasi antrian sesi permainan](#).
- Siapkan penskalaan kapasitas otomatis untuk memenuhi permintaan pemain yang diharapkan. Lihat [Menskalakan kapasitas hosting game dengan Amazon GameLift Servers](#).
- Siapkan armada siaga di armada lain Wilayah AWS dan ubah antrian dan penskalaan otomatis untuk menangani failover jika diperlukan.
- Siapkan alat observabilitas hosting, termasuk analitik dan pencatatan. Lihat [Pemantauan Amazon GameLift Servers](#).
- Otomatiskan penerapan Anda menggunakan [infrastruktur sebagai kode \(IAC\)](#). Lihat [Mengelola Amazon GameLift Servers sumber daya hosting menggunakan AWS CloudFormation](#).

Amazon GameLift Servers mendukung penggunaan AWS CloudFormation template untuk konfigurasi khusus penerapan apa pun. Anda juga dapat menggunakan AWS Cloud Development Kit (AWS CDK) untuk menentukan Amazon GameLift Servers sumber daya Anda. Untuk informasi selengkapnya tentang AWS CDK, lihat [Panduan AWS Cloud Development Kit \(AWS CDK\) Pengembang](#).

Untuk mengelola penyebaran AWS CloudFormation tumpukan Anda, sebaiknya gunakan alat dan layanan integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD) seperti AWS CodePipeline. Alat-alat ini membantu Anda menerapkan secara otomatis atau dengan persetujuan setiap kali Anda membangun biner server game. Dengan CI/CD alat atau layanan, penyebaran sumber daya untuk versi server game baru dapat terlihat seperti ini:

- Bangun dan uji biner server game Anda.
- Unggah biner ke Amazon GameLift Servers.
- Terapkan armada baru dengan build baru.
- Tambahkan armada baru ke antrian sesi game Anda dan hapus armada dengan versi build sebelumnya.

- Saat armada dengan build sebelumnya tidak lagi menghosting sesi game aktif, hapus AWS CloudFormation tumpukan armada tersebut.

Peta jalan pengembangan untuk hosting dengan kontainer Amazon GameLift Servers terkelola

Peta jalan ini memandu Anda melalui cara mengembangkan solusi hosting Amazon GameLift Servers terkelola untuk server game kontainer Anda. Kontainer terkelola hanyalah salah satu solusi hosting yang ditawarkan oleh Amazon GameLift Servers. Untuk informasi lebih lanjut tentang opsi hosting, lihat [Amazon GameLift Servers solusi](#).

Solusi kontainer terkelola dengan Amazon GameLift Servers memiliki komponen-komponen berikut:

- Satu atau beberapa armada kontainer, yang menggunakan instans Amazon Elastic Compute Cloud EC2 (Amazon) yang dioptimalkan untuk hosting game multipemain.
- Gambar kontainer dengan build server game Anda, diunggah ke repositori pribadi Amazon Elastic Container Registry (Amazon ECR). Build server game terintegrasi dengan SDK server untuk Amazon GameLift Servers dan dibangun untuk berjalan di Linux.
- Layanan backend yang berinteraksi dengan Amazon GameLift Servers layanan atas nama klien game Anda. Layanan backend menggunakan fungsionalitas dalam API layanan untuk Amazon GameLift Servers, yang merupakan bagian dari SDK AWS .
- Antrian sesi Amazon GameLift Servers permainan yang memproses permintaan untuk sesi permainan baru, mencari server game yang tersedia di semua armada, dan meminta server game untuk memulai sesi permainan.
- (Opsional) FlexMatch Mak comblang untuk membuat pertandingan multi-pemain dan mengatur sesi permainan untuk mereka.

Peta jalan ini menyajikan jalur yang disederhanakan untuk membuat server game kontainer Anda aktif dan berjalan dengan sukses dengan kontainer terkelola. Amazon GameLift Servers Setelah Anda memiliki komponen yang diperlukan, Anda dapat terus mengulangi pengembangan game dan menyesuaikan solusi hosting Anda. Saat Anda semakin dekat dengan peluncuran, lihat ini [Mempersiapkan game Anda untuk diluncurkan dengan Amazon GameLift Servers hosting](#) untuk bantuan dalam mempersiapkan solusi hosting Anda untuk penggunaan tingkat produksi.

i Percepat orientasi dengan alat ini untuk kontainer terkelola:

- [Kit starter kontainer](#) merampingkan integrasi dan pengaturan armada. Ini menambahkan fitur manajemen sesi permainan penting ke server game Anda, dan menggunakan templat yang telah dikonfigurasi sebelumnya untuk membangun armada kontainer dan pipeline penyebaran otomatis untuk server game Anda. Setelah penerapan, gunakan Amazon GameLift Servers konsol dan alat API untuk memantau kinerja armada, mengelola sesi game, dan menganalisis metrik.
- Untuk pengembang Unreal Engine dan Unity, gunakan [Amazon GameLift Serversplugin](#) untuk mengintegrasikan server game Anda dan membangun armada kontainer dari dalam lingkungan pengembangan mesin game Anda. Alur kerja terpandu plugin membantu Anda membuat solusi yang cepat dan sederhana dengan hosting berbasis cloud menggunakan kontainer terkelola. Anda dapat membangun fondasi ini untuk membuat solusi hosting khusus untuk game Anda.

Langkah 1: Siapkan server game Anda untuk dikerjakan Amazon GameLift Servers

Tambahkan fungsionalitas ke server game Anda sehingga dapat berkomunikasi dengan Amazon GameLift Servers layanan saat digunakan untuk hosting.

- Dapatkan SDK server untuk Amazon GameLift Servers (versi 5.2 atau lebih tinggi) untuk proyek game Anda. Server SDK tersedia dalam C ++, C #, dan Go. [Unduh SDK server untuk Amazon GameLift Servers](#). Server SDK tersedia dalam C ++, C #, dan Go.
- Ubah kode server game Anda untuk menambahkan fungsionalitas SDK server. Untuk panduan, lihat [Integrasikan game dengan server game khusus](#). Minimal, lakukan hal berikut:
 - Tambahkan kode untuk menginisialisasi Amazon GameLift Servers SDK dan membuat WebSocket koneksi dengan layanan. Amazon GameLift Servers Gunakan tindakan `InitSdk()` SDK server.
 - Tambahkan kode untuk melaporkan ke Amazon GameLift Servers layanan ketika proses server siap untuk meng-host sesi permainan. Gunakan tindakan `ProcessReady()` SDK server.
 - Menerapkan fungsi callback yang diperlukan `OnStartGameSession()`, dan `OnProcessTerminate()`. Dengan fungsi-fungsi ini, proses server game dapat mempertahankan koneksi dengan Amazon GameLift Servers layanan, memulai sesi permainan ketika diminta oleh Amazon GameLift Servers, dan menanggapi permintaan untuk mengakhiri proses server game.

- Tambahkan kode untuk melaporkan ke Amazon GameLift Servers layanan saat proses server mengakhiri sesi permainan. Gunakan tindakan `ProcessEnding()` SDK server.
- Package build server game Anda. Bangun server game Anda untuk berjalan di Linux. Siapkan build dan file lain yang diperlukan untuk menjalankan server game. Jika Anda mengembangkan pada Windows, langkah ini mungkin melibatkan pengaturan ruang kerja Linux terpisah atau menggunakan alat seperti subsistem Windows untuk Linux (WSL). Anda akan memerlukan lingkungan Linux untuk menguji build server game Anda, dan juga untuk membangun dan menguji gambar kontainer Anda.
- Uji integrasi server game Anda. Verifikasi bahwa server game terintegrasi Anda dapat terhubung ke Amazon GameLift Servers layanan dan menanggapi permintaan. Kami merekomendasikan untuk menyiapkan armada Amazon GameLift Servers Anywhere sederhana dengan workstation lokal sebagai host uji, seperti yang dijelaskan dalam [Siapkan pengujian lokal dengan Amazon GameLift Servers Anywhere](#). Instal build server game Anda ke host uji dan mulai proses server. Gunakan AWS CLI untuk meminta sesi permainan baru, dan verifikasi bahwa Amazon GameLift Servers layanan berhasil meminta proses server Anda untuk memulai sesi permainan.

Langkah 2: Siapkan klien game Anda untuk bergabung dengan sesi game yang dihosting

Buat cara bagi klien game Anda untuk meminta bergabung dengan sesi game, mendapatkan info koneksi, dan kemudian terhubung langsung ke sesi game yang dihosting. Pendekatan yang paling umum adalah mengatur fungsionalitas layanan backend yang berfungsi sebagai perantara antara klien game Anda dan layanan. Amazon GameLift Servers Pendekatan ini melindungi sumber daya hosting Anda dan memberi Anda kontrol yang lebih besar atas bagaimana pemain ditempatkan ke dalam sesi permainan.

- Bangun fungsionalitas layanan backend untuk hosting. Layanan backend berkomunikasi dengan Amazon GameLift Servers layanan dan memberikan informasi koneksi ke klien game. Fungsi ini termasuk memulai sesi permainan, menempatkan pemain ke dalam game, dan mengambil informasi sesi permainan. Untuk panduan, lihat [Integrasikan game dengan server game khusus](#). Minimal, lakukan hal berikut:
 - Dapatkan AWS SDK untuk Amazon GameLift Servers dan tambahkan ke proyek layanan backend Anda. Lihat [sumber daya Amazon GameLift Servers SDK untuk layanan klien](#).
 - Tambahkan kode untuk menginisialisasi Amazon GameLift Servers klien dan menyimpan pengaturan kunci. Lihat [Siapkan Amazon GameLift Servers pada layanan backend](#).

- Tambahkan fungsionalitas untuk memanggil aksi AWS SDK `CreateGameSession()` dan memberikan informasi koneksi sesi game ke klien game. Lihat [Membuat sesi permainan pada armada tertentu](#).

Memanggil `CreateGameSession()` adalah titik awal yang nyaman untuk meminta sesi permainan baru. Setelah Anda memiliki sistem penempatan sesi permainan di tempat (lihat Langkah 3), Anda akan mengganti kode ini dengan panggilan ke `StartGameSessionPlacement()` (atau `StartMatchmaking()` jika Anda menggunakan `FlexMatch`).

Untuk panduan tentang merancang layanan backend Anda, lihat. [Rancang layanan klien game Anda](#)

- Tambahkan fungsionalitas ke klien game Anda yang memungkinkan pemain bergabung dengan sesi game yang dihosting. Klien game membuat permintaan ke layanan backend Anda, tidak langsung ke Amazon GameLift Servers. Setelah layanan backend menyediakan informasi koneksi sesi game, klien game terhubung langsung dengan sesi game untuk memainkan game.
- Uji integrasi klien game Anda. Anda dapat menggunakan armada Amazon GameLift Servers Anywhere yang ada dengan workstation lokal untuk pengujian. Gunakan layanan backend baru untuk meminta sesi permainan baru, dan verifikasi bahwa: (1) Amazon GameLift Servers layanan berhasil meminta proses server Anda untuk memulai sesi permainan, dan (2) klien game dapat terhubung ke sesi permainan.

Langkah 3: Mengatur penempatan sesi permainan

Sesuaikan cara Anda Amazon GameLift Servers ingin memproses permintaan untuk sesi permainan baru dan temukan server game yang tersedia untuk meng-host mereka. Amazon GameLift Servers secara otomatis melacak ketersediaan semua server game di semua armada. Ketika klien game mengirim permintaan untuk bergabung dengan sesi permainan, Amazon GameLift Servers cari penempatan “terbaik” berdasarkan serangkaian prioritas yang ditentukan seperti latensi minimum, biaya, dan ketersediaan.

- Buat antrian sesi permainan untuk menempatkan sesi permainan baru dengan server game yang tersedia. Antrian adalah mekanisme utama untuk penempatan sesi permainan. Untuk panduan, lihat [Buat antrian sesi game](#).
- Minimal, tambahkan armada Anywhere Anda sebagai tujuan dalam antrian Anda. Semua pengaturan lainnya adalah kustomisasi opsional.

- Dalam kode layanan backend Anda, ubah **CreateGameSession()** panggilan ke **StartGameSessionPlacement()** Lihat [Membuat sesi permainan dalam antrean multi-lokasi](#).
- Buat mekanisme untuk memberi tahu klien game saat sesi permainan siap untuk bergabung. Saat dalam pengembangan, Anda dapat melakukan polling untuk status sesi game menggunakan panggilan ke **DescribeGameSessionPlacement**. Namun, sebelum menggunakan antrian untuk memproses volume tinggi, Anda harus mengaktifkan pemberitahuan acara. Lihat [Atur notifikasi kejadian untuk penempatan sesi game](#).
- Tambahkan FlexMatch perijodohan (opsional). Buat set aturan perijodohan dan buat konfigurasi perijodohan untuk bekerja dengan antrian sesi permainan Anda. Untuk panduan tentang pengaturan sistem perijodohan, lihat panduan [Amazon GameLift Servers FlexMatch pengembang](#).
- Uji sistem penempatan. Anda dapat menggunakan armada Amazon GameLift Servers Anywhere yang ada dengan workstation lokal untuk pengujian. Gunakan layanan backend untuk meminta sesi permainan baru, dan verifikasi bahwa Amazon GameLift Servers layanan berhasil meminta proses server Anda untuk memulai sesi permainan.

Langkah 4: Buat gambar kontainer server game

Setelah Anda berhasil mengintegrasikan server game Anda, buat gambar kontainer dengan server game Anda yang dapat dieksekusi. Simpan di repositori Amazon Elastic Container Registry (Amazon ECR) untuk digunakan. Amazon GameLift Servers Untuk petunjuk mendetail, lihat [Membangun gambar kontainer untuk Amazon GameLift Servers](#).

- Dapatkan template Dockerfile untuk wadah server game (disediakan oleh Amazon GameLift Servers). Ubah file untuk file build server game Anda.
- Bangun gambar kontainer server game. Bekerja di lingkungan Linux, gunakan alat Docker untuk membuat gambar Anda.
- Dorong gambar kontainer Anda ke Amazon ECR. Buat repositori publik atau pribadi di Amazon ECR, menggunakan yang sama Akun AWS dan Wilayah AWS tempat Anda berencana untuk menerapkan armada kontainer Anda. Dorong gambar kontainer Anda ke sana.
- Uji gambar kontainer Anda menggunakan armada Anywhere Anda (opsional). Anda mungkin ingin menguji gambar kontainer secara lokal sebelum menerapkannya ke armada kontainer yang dihosting cloud. Anda dapat menggunakan armada Amazon GameLift Servers Anywhere yang ada dengan workstation lokal untuk pengujian. Instal dan jalankan wadah server game dan verifikasi bahwa: (1) Amazon GameLift Servers layanan berhasil meminta proses server Anda untuk memulai sesi permainan, dan (2) klien game dapat terhubung ke sesi permainan.

Langkah 5: Buat armada kontainer berbasis cloud

Hingga saat ini Anda telah bekerja dengan armada Anywhere yang dikelola sendiri untuk menguji dan mengulangi komponen game Anda. Bagian terakhir dari solusi Anda adalah menyiapkan sumber daya hosting berbasis cloud yang Anda perlukan untuk sistem produksi. Untuk mulai merencanakan dan mengonfigurasi produksi, Anda ingin menyiapkan armada kontainer Amazon GameLift Servers terkelola dan menyesuaikannya untuk produksi.

- Buat definisi grup kontainer. Definisi grup kontainer menggambarkan arsitektur kontainer untuk sebuah armada, dan mengidentifikasi gambar kontainer mana yang akan digunakan. Lihat [Buat definisi grup kontainer untuk armada Amazon GameLift Servers kontainer](#). Buat definisi grup kontainer Anda di Wilayah AWS tempat yang sama di mana gambar kontainer disimpan. Untuk informasi lebih lanjut tentang memilih lokasi armada, lihat [Lokasi armada](#). Minimal, lakukan hal berikut:
 - Buat definisi grup kontainer server game.
 - Tambahkan definisi kontainer dengan gambar kontainer dengan build server game Anda.
 - Konfigurasi rentang port untuk proses server game kontainer.
- Buat armada kontainer terkelola. Saat Anda membuat armada, Amazon GameLift Servers segera mulai menerapkan build server game Anda untuk hosting. Anda dapat mengonfigurasi banyak aspek armada yang dikelola. Untuk panduan, lihat [Buat sebuah Amazon GameLift Servers armada kontainer terkelola](#). Minimal, lakukan hal berikut:
 - Siapkan peran layanan AWS Identity and Access Management (IAM) untuk armada kontainer. Lihat [Menyiapkan peran layanan IAM untuk Amazon GameLift Servers](#).
 - Tentukan definisi grup kontainer server game yang akan diterapkan ke instance armada.
 - Gunakan nilai default jika tersedia untuk semua parameter lainnya. Amazon GameLift Servers menghitung beberapa parameter untuk konfigurasi optimal.
- Tambahkan armada kontainer ke antrian Anda. Dalam antrian sesi permainan Anda, ganti armada uji Anywhere dengan armada kontainer terkelola Anda.
- Uji hosting game dengan armada kontainer Anda. Pada titik ini Anda harus dapat menguji seluruh solusi. Mulai klien game dan minta sesi permainan melalui layanan backend. Dapatkan info koneksi dan sambungkan ke sesi permainan di armada kontainer.
- Iterasi pada penerapan armada Anda. Anda dapat memperbarui definisi grup kontainer dan konfigurasi armada, lalu menerapkan pembaruan ke armada yang ada.

Langkah 6: Sesuaikan solusi kontainer terkelola Anda

Saat Anda mempersiapkan peluncuran game, Anda harus menyempurnakan sumber daya hosting terkelola Anda. Beberapa keputusan yang perlu dipertimbangkan meliputi:

- Optimalkan konfigurasi armada kontainer Anda. Lihat [Sesuaikan armada Amazon GameLift Servers kontainer](#).
- Pertimbangkan untuk menambahkan armada Spot untuk penghematan biaya. Lihat [Tutorial: Membuat Amazon GameLift Servers antrian dengan Instans Spot](#).
- Jika server game Anda perlu mengkomunikasikan AWS sumber daya lain, siapkan peran IAM untuk mengelola akses. Lihat [Berkomunikasi dengan AWS sumber daya lain dari armada Anda](#).
- Tentukan di mana secara geografis Anda ingin memposisikan server game. Tambahkan lokasi jarak jauh ke armada terkelola Anda. Lihat [Sesuaikan Amazon GameLift Servers EC2 armada yang dikelola](#).
- Bereksperimenlah dengan opsi penempatan sesi game untuk armada terkelola, termasuk menyesuaikan pengaturan prioritas. Lihat [Kustomisasi antrian sesi permainan](#).
- Siapkan penskalaan kapasitas otomatis untuk memenuhi permintaan pemain yang diharapkan. Lihat [Menskalakan kapasitas hosting game dengan Amazon GameLift Servers](#).
- Buat armada lain Wilayah AWS dan modifikasi antrian dan penskalaan otomatis untuk menangani failover sesuai kebutuhan.
- Siapkan alat observabilitas hosting, termasuk analitik dan pencatatan. Lihat [Pemantauan Amazon GameLift Servers](#).
- Otomatiskan penerapan armada Anda menggunakan [infrastruktur sebagai kode \(IaC\)](#). Lihat [Mengelola Amazon GameLift Servers sumber daya hosting menggunakan AWS CloudFormation](#).

Amazon GameLift Servers mendukung penggunaan AWS CloudFormation template untuk konfigurasi khusus penerapan apa pun. Anda juga dapat menggunakan AWS Cloud Development Kit (AWS CDK) untuk menentukan Amazon GameLift Servers sumber daya Anda. Untuk informasi selengkapnya tentang AWS CDK, lihat [Panduan AWS Cloud Development Kit \(AWS CDK\) Pengembang](#).

Untuk mengelola penyebaran AWS CloudFormation tumpukan Anda, sebaiknya gunakan alat dan layanan integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD) seperti AWS CodePipeline. Alat-alat ini membantu Anda menerapkan secara otomatis atau dengan persetujuan setiap kali Anda membangun biner server game. Dengan CI/CD alat atau layanan, penyebaran sumber daya untuk versi server game baru dapat terlihat seperti ini:

- Bangun dan uji biner server game Anda.
- Unggah biner ke Amazon GameLift Servers.
- Terapkan armada baru dengan build baru.
- Tambahkan armada baru ke antrian sesi game Anda dan hapus armada dengan versi build sebelumnya.
- Saat armada dengan build sebelumnya tidak lagi menghosting sesi game aktif, hapus AWS CloudFormation tumpukan armada tersebut.

Peta jalan pengembangan untuk hosting dengan Amazon GameLift Servers Anywhere

Peta jalan ini memandu Anda melalui cara mengembangkan solusi hosting untuk game multipemain Anda untuk digunakan dengan sumber daya Anda sendiri (perangkat keras lokal atau mesin virtual). Amazon GameLift Servers menawarkan beberapa opsi hosting game; untuk informasi lebih lanjut tentang opsi ini, lihat [Amazon GameLift Servers solusi](#).

Dengan hosting Amazon GameLift Servers Anywhere, server game Anda di-host pada sumber daya komputasi yang Anda suplai dan kelola. Anda dapat membuat armada Anywhere dengan konfigurasi yang Anda butuhkan dan secara geografis terletak di mana pun pemain Anda berada. Amazon GameLift Servers memberikan fitur-fitur berikut untuk armada Anywhere:

- Menangani proses penempatan sesi permainan untuk Anda berdasarkan konfigurasi Anda, termasuk:
 - Melacak ketersediaan server game di seluruh armada Anywhere Anda.
 - Memproses permintaan game dari layanan klien game Anda dan mencocokkan permintaan game dengan server yang tersedia.
 - Meminta server game di armada Anywhere untuk memulai sesi permainan.
 - Mengkomunikasikan detail koneksi kembali ke klien game.
- Mengumpulkan metrik kinerja untuk proses penempatan sesi dan metrik penggunaan untuk sesi permainan dan pemain.
- Mendukung set fitur FlexMatch perjodohan lengkap, sehingga Anda dapat membangun mak comblang dan mengintegrasikannya dengan sistem penempatan sesi permainan.
- Menawarkan Amazon GameLift Servers Agen untuk menangani tugas manajemen host utama pada armada Anywhere.

- Mendukung penggabungan dengan armada Amazon GameLift Servers terkelola untuk solusi hybrid yang fleksibel.

Solusi Amazon GameLift Servers Anywhere terdiri dari komponen-komponen berikut:

- Satu atau beberapa armada Amazon GameLift Servers Anywhere dengan sumber daya hosting lokal atau lainnya, dikelola dengan manajemen konfigurasi dan perkakas penerapan yang ada. (Anda dapat secara opsional menggunakan AWS Systems Manager.)
- Sebuah build server game, terintegrasi dengan SDK server untuk Amazon GameLift Servers, untuk digunakan di semua armada.
- Klien game dan layanan backend, terintegrasi dengan AWS SDK, untuk berinteraksi dengan Amazon GameLift Servers layanan dan meminta sesi permainan.
- Amazon GameLift Servers Antrian untuk menempatkan sesi permainan baru dengan server game yang tersedia di semua armada.
- (Opsional) FlexMatch Mak comblang untuk membuat pertandingan multi-pemain dan mengatur sesi permainan untuk mereka.

Peta jalan ini menyajikan jalur yang efisien untuk membuat game multipemain Anda aktif dan berjalan dengan sukses dengan Amazon GameLift Servers hosting Anywhere. Setelah Anda memiliki komponen yang diperlukan, Anda dapat terus mengulangi pengembangan game dan menyesuaikan solusi hosting Anda. Saat Anda semakin dekat dengan peluncuran, lihat ini [Mempersiapkan game Anda untuk diluncurkan dengan Amazon GameLift Servers hosting](#) untuk bantuan dalam mempersiapkan solusi hosting Anda untuk penggunaan tingkat produksi.

i Mulailah dengan Amazon GameLift Servers plugin untuk Unreal Engine dan Unity

Untuk penyebaran yang lebih cepat, coba [Amazon GameLift Servers plugin](#) untuk Unreal Engine dan Unity. Ini menyediakan alur kerja UI terpandu untuk menyebarkan server game Anda dengan cepat dengan pengaturan minimal, sehingga Anda dapat mencoba komponen game Anda dalam aksi. Kemudian Anda dapat membangun fondasi ini untuk membuat solusi hosting khusus untuk game Anda. Untuk detail selengkapnya, lihat [Jelajahi dengan Amazon GameLift Servers plugin](#).

Langkah 1: Siapkan server game Anda untuk dikerjakan Amazon GameLift Servers

Tambahkan fungsionalitas ke server game Anda sehingga dapat berkomunikasi dengan Amazon GameLift Servers layanan saat digunakan untuk hosting.

- Dapatkan SDK server untuk Amazon GameLift Servers (versi 5.x) untuk proyek game Anda. Server SDK tersedia dalam C ++, C #, dan Go. [Unduh SDK Amazon GameLift Servers server](#).
- Ubah kode server game Anda untuk menambahkan fungsionalitas SDK server. Untuk panduan, lihat [Integrasikan game dengan server game khusus](#). Minimal, lakukan hal berikut:
 - Tambahkan kode untuk menginisialisasi Amazon GameLift Servers SDK dan membuat WebSocket koneksi dengan layanan. Amazon GameLift Servers Gunakan tindakan SDK server `InitSdk()` dan sertakan parameter server, yang diperlukan untuk armada Anywhere.
 - Tambahkan kode untuk melaporkan ke Amazon GameLift Servers layanan ketika proses server siap untuk meng-host sesi permainan. Gunakan tindakan `ProcessReady()` SDK server.
 - Menerapkan fungsi callback yang diperlukan `OnProcessTerminate()`, dan `OnStartGameSession()`. Dengan fungsi-fungsi ini, proses server game dapat mempertahankan koneksi dengan Amazon GameLift Servers layanan, memulai sesi permainan ketika diminta oleh Amazon GameLift Servers, dan menanggapi prompt untuk mengakhiri proses server game.
 - Tambahkan kode untuk melaporkan ke Amazon GameLift Servers layanan saat proses server mengakhiri sesi permainan. Gunakan tindakan `ProcessEnding()` SDK server.
- Package build server game Anda. Buat skrip penginstalan dengan file build, dependensi, dan perangkat lunak pendukung Anda. Lihat [Membuat paket file bangunan game Anda](#). Sebaiknya gunakan bucket Amazon Simple Storage Service (Amazon S3) untuk menyimpan versi build game Anda.
- Uji integrasi server game Anda. Untuk tugas ini, kami sarankan menyiapkan armada Amazon GameLift Servers Anywhere untuk workstation lokal, seperti yang dijelaskan dalam [Siapkan pengujian lokal dengan Amazon GameLift Servers Anywhere](#). Untuk langkah ini, instal server game Anda secara manual ke perangkat uji dan mulai proses server. Gunakan AWS CLI untuk meminta sesi permainan baru, dan verifikasi bahwa Amazon GameLift Servers layanan berhasil meminta proses server Anda untuk memulai sesi permainan.

Langkah 2: Siapkan klien game Anda untuk bergabung dengan sesi game yang dihosting

Buat cara bagi klien game Anda untuk meminta bergabung dengan sesi game, mendapatkan info koneksi, dan kemudian terhubung langsung ke sesi game yang dihosting. Pendekatan yang paling umum adalah mengatur fungsionalitas layanan backend yang berfungsi sebagai perantara antara klien game Anda dan layanan. Amazon GameLift Servers Pendekatan ini melindungi sumber daya hosting Anda dan memberi Anda kontrol yang lebih besar atas bagaimana pemain ditempatkan ke dalam sesi permainan.

- Bangun fungsionalitas layanan backend untuk hosting. Layanan backend berkomunikasi dengan Amazon GameLift Servers layanan dan memberikan informasi koneksi ke klien game. Fungsionalitas ini termasuk memulai sesi permainan, menempatkan pemain ke dalam game, dan mengambil informasi sesi permainan. Untuk panduan, lihat [Integrasikan game dengan server game khusus](#). Minimal, lakukan hal berikut:
 - Dapatkan AWS SDK untuk Amazon GameLift Servers dan tambahkan ke proyek layanan backend Anda. Lihat [sumber daya Amazon GameLift Servers SDK untuk layanan klien](#).
 - Tambahkan kode untuk menginisialisasi Amazon GameLift Servers klien dan menyimpan pengaturan kunci. Lihat [Siapkan Amazon GameLift Servers pada layanan backend](#).
 - Tambahkan fungsionalitas untuk memanggil aksi AWS SDK `CreateGameSession()` dan memberikan informasi koneksi sesi game ke klien game. Lihat [Membuat sesi permainan pada armada tertentu](#).

Memanggil `CreateGameSession()` adalah titik awal yang nyaman untuk meminta sesi permainan baru. Setelah Anda memiliki sistem penempatan sesi permainan di tempat (lihat Langkah 3), Anda akan mengganti kode ini dengan panggilan ke `StartGameSessionPlacement()` (atau `StartMatchmaking()` jika Anda menggunakan `FlexMatch`).

Untuk panduan tentang merancang layanan backend Anda, lihat. [Rancang layanan klien game Anda](#)

- Tambahkan fungsionalitas ke klien game Anda yang memungkinkan pemain bergabung dengan sesi game yang dihosting. Klien game membuat permintaan ke layanan backend Anda, tidak langsung ke Amazon GameLift Servers. Setelah layanan backend menyediakan informasi koneksi sesi game, klien game terhubung langsung dengan sesi game untuk memainkan game.
- Uji integrasi klien game Anda. Anda dapat menggunakan armada Amazon GameLift Servers Anywhere yang sama dengan workstation lokal untuk pengujian.

Langkah 3: Mengatur penempatan sesi permainan

Sesuaikan cara Anda Amazon GameLift Servers ingin memproses permintaan untuk sesi permainan baru dan temukan server game yang tersedia untuk meng-host mereka. Amazon GameLift Servers secara otomatis melacak ketersediaan semua server game di semua armada. Ketika klien game mengirim permintaan untuk bergabung dengan sesi permainan, Amazon GameLift Servers cari penempatan “terbaik” berdasarkan serangkaian prioritas yang ditentukan seperti latensi minimum, biaya, dan ketersediaan.

- Buat antrian sesi permainan untuk menempatkan sesi permainan baru dengan server game yang tersedia. Antrian adalah mekanisme utama untuk penempatan sesi permainan. Untuk panduan, lihat [Buat antrian sesi game](#).
- Minimal, tambahkan armada Anywhere Anda sebagai tujuan dalam antrian Anda. Semua pengaturan lainnya adalah opsional.
- Dalam kode layanan backend Anda, ubah **CreateGameSession()** panggilan ke **StartGameSessionPlacement()** Lihat [Membuat sesi permainan dalam antrian multi-lokasi](#).
- Buat mekanisme untuk memberi tahu klien game saat sesi permainan siap untuk bergabung. Saat dalam pengembangan, Anda dapat melakukan polling untuk status sesi game menggunakan panggilan ke DescribeGameSessionPlacement. Namun, sebelum menggunakan antrian untuk memproses volume tinggi, Anda harus mengaktifkan notifikasi acara. Lihat [Atur notifikasi kejadian untuk penempatan sesi game](#).
- (Opsional) Tambahkan komponen FlexMatch perijodohan. Untuk panduan, lihat [panduan Amazon GameLift Servers FlexMatch pengembang](#).

Langkah 4: Siapkan armada Anywhere dengan Amazon GameLift Servers Agen

Hingga saat ini Anda telah bekerja dengan perangkat lokal (terdaftar sebagai komputasi armada Anywhere) untuk menguji dan mengulangi komponen game Anda. Langkah selanjutnya adalah mengatur jenis armada yang Anda perlukan untuk sistem produksi. Untuk sumber daya ini, gunakan Amazon GameLift Servers Agen untuk mengelola beberapa tugas manajemen host on-compute utama. Untuk detail selengkapnya, lihat [Bekerja dengan Amazon GameLift Servers Agen](#).

- Dapatkan Amazon GameLift Servers Agen dan tambahkan ke paket instalasi server game Anda. Dapatkan dan buat kode sumber Agen, tersedia di repositori [Amazon GameLift Servers Agen Github](#). Tempatkan file JAR yang dihasilkan yang dapat dieksekusi ke direktori yang sama dengan game build yang dapat dieksekusi.

- Ubah skrip startup Anda untuk Agen sesuai kebutuhan. Pastikan Agen yang dapat dieksekusi diluncurkan segera setelah komputasi mulai berjalan. Lihat file readme di repo Agen untuk bantuan menginstal dan menjalankan Agen pada komputasi hosting Anda. Perintah peluncuran Anda harus menyertakan opsi untuk menentukan, setidaknya, ID armada Anywhere dan Wilayah AWS, lokasi khusus, dan nama komputasi.

Agen secara otomatis menangani tugas-tugas berikut untuk Anda, jadi jika Anda telah menangani tugas ini dengan skrip, Anda dapat menghapusnya:

- Panggilan `RegisterCompute()` untuk menambahkan komputasi ke armada Anywhere.
- Panggilan `GetComputeAuthToken()` untuk mengautentikasi server game saat mereka terhubung ke Amazon GameLift Servers layanan. Agen mengelola mendapatkan dan menyegarkan token autentikasi, yang dapat digunakan oleh semua proses server game yang berjalan pada komputasi.
- Memulai proses server baru pada komputasi berdasarkan serangkaian instruksi runtime.
- Buat konfigurasi runtime untuk komputasi di armada Anywhere Anda. Anda dapat menggunakan Amazon GameLift Servers konsol atau AWS CLI untuk membuat atau memodifikasi instruksi runtime untuk armada. Agen melaksanakan instruksi ini dan secara berkala meminta pembaruan dari Amazon GameLift Servers layanan.
- Siapkan atau ubah antrian sesi permainan Anda sesuai kebutuhan. Buat antrian baru (atau perbarui yang sudah ada) untuk menggunakan armada Anywhere yang disebarkan dengan Agen. Amazon GameLift Servers
- Uji integrasi Agen dengan armada Anywhere Anda. Periksa apakah Agen memulai proses server dengan benar berdasarkan konfigurasi runtime.

Langkah 5: Sesuaikan armada Anywhere Anda

Saat Anda mempersiapkan peluncuran game, Anda harus menyempurnakan sumber daya hosting terkelola Anda. Beberapa keputusan yang perlu dipertimbangkan meliputi:

- Otomatiskan proses memulai dan mematikan komputasi sesuai kebutuhan, termasuk menginstal dan menjalankan perangkat lunak server game. Komputasi daur ulang berguna untuk memastikan bahwa mereka diperbarui secara berkala, dan mematikan komputasi dapat menghemat biaya saat tidak diperlukan.
- Jika server game Anda perlu mengkomunikasikan AWS sumber daya lain, siapkan peran IAM untuk mengelola akses. Lihat [Berkomunikasi dengan AWS sumber daya lain dari armada Anda](#).

- Tentukan di mana secara geografis Anda ingin memposisikan server game. Tambahkan lokasi jarak jauh ke armada terkelola Anda. Lihat [Sesuaikan Amazon GameLift Servers EC2 armada yang dikelola](#).
- Optimalkan kinerja armada dengan memilih konfigurasi sumber daya komputasi, lalu konfigurasi instruksi runtime untuk menjalankan jumlah optimal proses server per komputasi.
- Bereksperimenlah dengan opsi penempatan sesi game untuk armada terkelola, termasuk menyesuaikan pengaturan prioritas. Lihat [Kustomisasi antrian sesi permainan](#).
- Buat mekanisme untuk menangani penskalaan kapasitas manual atau otomatis untuk memenuhi permintaan pemain yang diharapkan. Pertimbangkan faktor apa yang harus mendorong sistem untuk menambah atau mengurangi jumlah komputasi yang tersedia untuk menyelenggarakan sesi permainan.
- Rancang dan implementasikan failover ke sumber daya lain jika diperlukan.
- Siapkan alat observabilitas hosting, termasuk analitik dan pencatatan. Lihat [Pemantauan Amazon GameLift Servers](#).

Peta jalan pengembangan untuk hosting hybrid dengan Amazon GameLift Servers

Peta jalan ini memandu Anda melalui cara mengembangkan solusi hosting untuk game multipemain Anda. Amazon GameLift Servers menawarkan beberapa opsi hosting game; untuk informasi lebih lanjut tentang opsi ini, lihat [Amazon GameLift Servers solusi](#).

Solusi hybrid menggunakan kombinasi sumber daya hosting, termasuk sumber daya berbasis cloud yang dikelola oleh Amazon GameLift Servers dan sumber daya hosting yang dikelola sendiri. Untuk diskusi lebih rinci tentang hosting hybrid, lihat artikel ini: [Hosting server game hybrid dengan Amazon GameLift Servers Anywhere](#). Dengan Amazon GameLift Servers, Anda dapat menyiapkan solusi hybrid yang menggunakan komponen dan proses umum, sehingga Anda dapat mengelola armada global secara terpusat dan dengan mudah memindahkan beban di antara semua jenis sumber daya.

Arsitektur hybrid terdiri dari komponen-komponen berikut:

- Satu atau beberapa armada Amazon GameLift Servers terkelola, yang menggunakan instans Amazon Elastic Compute Cloud EC2 (Amazon) yang dioptimalkan untuk hosting game multipemain.

- Satu atau beberapa armada Amazon GameLift Servers Anywhere, yang menggunakan sumber daya hosting lokal atau lainnya yang ada, termasuk manajemen konfigurasi dan perkakas penerapan. (Anda dapat secara opsional menggunakan AWS Systems Manager.)
- Sebuah build server game tunggal, terintegrasi dengan SDK server untuk Amazon GameLift Servers, untuk diterapkan di semua armada.
- Klien game tunggal dan layanan backend, terintegrasi dengan AWS SDK, untuk berinteraksi dengan Amazon GameLift Servers layanan dan meminta sesi permainan.
- Amazon GameLift Servers Antrian bersama untuk menempatkan sesi permainan baru dengan server game yang tersedia dan beban keseimbangan di semua armada.
- Amazon GameLift Servers Agen, yang digunakan dengan armada Anywhere, untuk menyederhanakan tugas manajemen proses server di seluruh komputasi di semua armada.
- (Opsional) FlexMatch Mak comblang untuk membuat pertandingan multi-pemain dan mengatur sesi permainan untuk mereka.

Peta jalan ini menyajikan jalur yang efisien untuk membuat game multipemain Anda aktif dan berjalan dengan sukses dalam solusi hosting hybrid. Amazon GameLift Servers Setelah Anda memiliki komponen yang diperlukan, Anda dapat terus mengulangi pengembangan game dan menyesuaikan solusi hosting Anda. Saat Anda semakin dekat dengan peluncuran, lihat ini [Mempersiapkan game Anda untuk diluncurkan dengan Amazon GameLift Servers hosting](#) untuk bantuan dalam mempersiapkan solusi hosting Anda untuk penggunaan tingkat produksi.

Mulailah dengan Amazon GameLift Servers plugin

Jika Anda mengembangkan proyek dengan Unreal Engine atau Unity, mulailah menyiapkan game Anda untuk hosting dengan Amazon GameLift Servers plugin. Dengan plugin ini, Anda dapat menambahkan Amazon GameLift Servers SDKs ke proyek game Anda dan menggunakan alur kerja yang dipandu untuk membangun versi kerja sederhana dari solusi hosting hybrid dengan armada Anywhere dan armada Amazon GameLift Servers terkelola. Anda kemudian dapat menggunakan dasar-dasar ini untuk membangun dan menyesuaikan sesuai kebutuhan.

Langkah 1: Siapkan server game Anda untuk bekerja dengan Amazon GameLift Servers

Tambahkan fungsionalitas ke server game Anda sehingga dapat berkomunikasi dengan Amazon GameLift Servers layanan saat digunakan untuk hosting. Fungsionalitas yang sama diperlukan jika server game berjalan pada armada Amazon GameLift Servers terkelola atau armada Anywhere.

- Dapatkan SDK server untuk Amazon GameLift Servers (versi 5.x) untuk proyek game Anda. Server SDK tersedia dalam C ++, C #, dan Go. [Unduh SDK server untuk Amazon GameLift Servers](#).
- Ubah kode server game Anda untuk menambahkan fungsionalitas SDK server. Untuk panduan, lihat [Integrasikan game dengan server game khusus](#). Minimal, lakukan hal berikut:
 - Tambahkan kode untuk menginisialisasi Amazon GameLift Servers SDK dan membuat WebSocket koneksi dengan layanan. Amazon GameLift Servers Gunakan tindakan `InitSdk()` SDK server. Sertakan kode untuk menentukan parameter server saat berjalan pada komputasi armada Anywhere.
 - Tambahkan kode untuk melaporkan ke Amazon GameLift Servers layanan ketika proses server siap untuk meng-host sesi permainan. Gunakan tindakan `ProcessReady()` SDK server.
 - Menerapkan fungsi callback yang diperlukan `OnProcessTerminate()`, dan `OnStartGameSession()`. Dengan fungsi-fungsi ini, proses server game dapat mempertahankan koneksi dengan Amazon GameLift Servers layanan, memulai sesi permainan ketika diminta oleh Amazon GameLift Servers, dan menanggapi prompt untuk mengakhiri proses server game.
 - Tambahkan kode untuk melaporkan ke Amazon GameLift Servers layanan saat proses server mengakhiri sesi permainan. Gunakan tindakan `ProcessEnding()` SDK server.
- Package build server game Anda. Buat skrip penginstalan dengan file build, dependensi, dan perangkat lunak pendukung Anda. Lihat [Membuat paket file bangunan game Anda](#). Sebaiknya gunakan bucket Amazon Simple Storage Service (Amazon S3) untuk menyimpan versi build game Anda.
- Uji integrasi server game Anda. Untuk tugas ini, kami sarankan menyiapkan armada Amazon GameLift Servers Anywhere dengan workstation lokal, seperti yang dijelaskan dalam [Siapkan pengujian lokal dengan Amazon GameLift Servers Anywhere](#). Untuk langkah ini, instal server game Anda secara manual ke perangkat uji dan mulai proses server. Gunakan AWS CLI untuk meminta sesi permainan baru, dan verifikasi bahwa Amazon GameLift Servers layanan berhasil meminta proses server Anda untuk memulai sesi permainan.

Langkah 2: Siapkan klien game Anda untuk bergabung dengan sesi game yang dihosting

Buat cara bagi klien game Anda untuk meminta bergabung dengan sesi game, mendapatkan info koneksi, dan kemudian terhubung langsung ke sesi game yang dihosting. Pendekatan yang paling umum adalah mengatur fungsionalitas layanan backend yang berfungsi sebagai perantara antara klien game Anda dan layanan. Amazon GameLift Servers Pendekatan ini melindungi sumber daya hosting Anda dan memberi Anda kontrol yang lebih besar atas bagaimana pemain ditempatkan ke dalam sesi permainan.

- Bangun fungsionalitas layanan backend untuk hosting. Layanan backend berkomunikasi dengan Amazon GameLift Servers layanan dan memberikan informasi koneksi ke klien game. Fungsionalitas ini termasuk memulai sesi permainan, menempatkan pemain ke dalam game, dan mengambil informasi sesi permainan. Untuk panduan, lihat [Integrasikan game dengan server game khusus](#). Minimal, lakukan hal berikut:
 - Dapatkan AWS SDK untuk Amazon GameLift Servers dan tambahkan ke proyek layanan backend Anda. Lihat [sumber daya Amazon GameLift Servers SDK untuk layanan klien](#).
 - Tambahkan kode untuk menginisialisasi Amazon GameLift Servers klien dan menyimpan pengaturan kunci. Lihat [Siapkan Amazon GameLift Servers pada layanan backend](#).
 - Tambahkan fungsionalitas untuk memanggil aksi AWS SDK `CreateGameSession()` dan memberikan informasi koneksi sesi game ke klien game. Lihat [Membuat sesi permainan pada armada tertentu](#).

Memanggil `CreateGameSession()` adalah titik awal yang nyaman untuk meminta sesi permainan baru. Setelah Anda memiliki sistem penempatan sesi permainan di tempat (lihat Langkah 3), Anda akan mengganti kode ini dengan panggilan ke `StartGameSessionPlacement()` (atau `StartMatchmaking()` jika Anda menggunakan `FlexMatch`).

Untuk panduan tentang merancang layanan backend Anda, lihat. [Rancang layanan klien game Anda](#)

- Tambahkan fungsionalitas ke klien game Anda yang memungkinkan pemain bergabung dengan sesi game yang dihosting. Klien game membuat permintaan ke layanan backend Anda, tidak langsung ke Amazon GameLift Servers. Setelah layanan backend menyediakan informasi koneksi sesi game, klien game terhubung langsung dengan sesi game untuk memainkan game.
- Uji integrasi klien game Anda. Anda dapat menggunakan armada Amazon GameLift Servers Anywhere yang sama dengan workstation lokal untuk pengujian.

Selama fase pengembangan, jika Anda ingin menguji perilaku build game dalam armada Amazon GameLift Servers terkelola, sebaiknya Anda juga menyiapkan lingkungan pengujian [berbasis cloud](#). Solusi Amazon GameLift Servers Toolkit ini meniru perilaku armada terkelola tetapi memungkinkan Anda memperbarui build server game dengan waktu penyelesaian minimal.

Langkah 3: Mengatur penempatan sesi permainan

Sesuaikan cara Anda Amazon GameLift Servers ingin memproses permintaan untuk sesi permainan baru dan temukan server game yang tersedia untuk meng-host mereka. Amazon GameLift Servers secara otomatis melacak ketersediaan semua server game di semua armada. Ketika klien game mengirim permintaan untuk bergabung dengan sesi permainan, Amazon GameLift Servers cari penempatan “terbaik” berdasarkan serangkaian prioritas yang ditentukan seperti latensi minimum, biaya, dan ketersediaan.

- Buat antrian sesi permainan untuk menempatkan sesi permainan baru dengan server game yang tersedia. Antrian adalah mekanisme utama untuk penempatan sesi permainan. Untuk panduan, lihat [Buat antrian sesi game](#).
- Minimal, tambahkan armada Anywhere Anda sebagai tujuan dalam antrian Anda. Semua pengaturan lainnya adalah kustomisasi opsional.
- Dalam kode layanan backend Anda, ubah **CreateGameSession()** panggilan ke **StartGameSessionPlacement()** Lihat [Membuat sesi permainan dalam antrian multi-lokasi](#).
- Buat mekanisme untuk memberi tahu klien game saat sesi permainan siap untuk bergabung. Saat dalam pengembangan, Anda dapat melakukan polling untuk status sesi game menggunakan panggilan ke DescribeGameSessionPlacement. Namun, sebelum menggunakan antrian untuk memproses volume tinggi, Anda harus mengaktifkan notifikasi acara. Lihat [Atur notifikasi kejadian untuk penempatan sesi game](#).
- (Opsional) Tambahkan komponen FlexMatch perjudian. Untuk panduan, lihat [panduan Amazon GameLift Servers FlexMatch pengembang](#).

Langkah 4: Siapkan armada Anywhere dengan Amazon GameLift Servers Agen

Hingga saat ini Anda telah bekerja dengan perangkat lokal (terdaftar sebagai komputasi armada Anywhere) untuk menguji dan mengulangi komponen game Anda. Langkah selanjutnya adalah mengatur jenis armada yang Anda perlukan untuk sistem produksi. Mulailah dengan armada Anywhere, dan tambahkan Amazon GameLift Servers Agen untuk mengelola beberapa tugas

manajemen host on-compute utama. Untuk detail selengkapnya, lihat [Bekerja dengan Amazon GameLift Servers Agen](#).

- Dapatkan Amazon GameLift Servers Agen dan tambahkan ke paket instalasi server game Anda. Dapatkan dan buat kode sumber Agen, tersedia di repositori [Amazon GameLift Servers Agen Github](#). Tempatkan file JAR yang dihasilkan yang dapat dieksekusi ke direktori yang sama dengan game build yang dapat dieksekusi.
- Ubah skrip startup Anda untuk Agen sesuai kebutuhan. Pastikan Agen yang dapat dieksekusi diluncurkan segera setelah komputasi mulai berjalan. Lihat file readme di repo Agen untuk bantuan menginstal dan menjalankan Agen pada komputasi hosting Anda. Perintah peluncuran Anda harus menyertakan opsi untuk menentukan, setidaknya, ID armada Anywhere dan Wilayah AWS, lokasi khusus, dan nama komputasi.

Agen secara otomatis menangani tugas-tugas berikut untuk Anda, jadi jika Anda telah menangani tugas ini dengan skrip, Anda dapat menghapusnya:

- Panggilan `RegisterCompute()` untuk menambahkan komputasi ke armada Anywhere.
- Panggilan `GetComputeAuthToken()` untuk mengautentikasi server game saat mereka terhubung ke Amazon GameLift Servers layanan. Agen mengelola mendapatkan dan menyegarkan token autentikasi, yang dapat digunakan oleh semua proses server game yang berjalan pada komputasi.
- Memulai proses server baru pada komputasi berdasarkan serangkaian instruksi runtime.
- Buat konfigurasi runtime untuk komputasi di armada Anywhere Anda. Minimal, tentukan jalur peluncuran untuk server game Anda yang dapat dieksekusi. Anda dapat menggunakan Amazon GameLift Servers konsol atau AWS CLI untuk membuat atau memodifikasi instruksi runtime untuk armada. Agen melaksanakan instruksi ini dan secara berkala meminta pembaruan dari Amazon GameLift Servers layanan.
- Siapkan atau ubah antrian sesi permainan Anda sesuai kebutuhan. Buat antrian baru (atau perbarui yang sudah ada) dan tentukan tujuan untuk armada Anywhere yang digunakan dengan Agen. Amazon GameLift Servers
- Uji integrasi Agen dengan armada Anywhere Anda. Periksa apakah Agen memulai proses server dengan benar berdasarkan konfigurasi runtime.

Langkah 5: Buat armada berbasis cloud terkelola

Buat armada Amazon GameLift Servers terkelola untuk melengkapi EC2 armada Anywhere Anda. Jika Anda menyiapkan lingkungan pengujian berbasis cloud di Langkah 2 untuk mempercepat

pengembangan, rencanakan untuk membuat armada terkelola setelah Anda menyelesaikan sebagian besar pengembangan dan pengujian game. Anda memerlukan armada yang dikelola sepenuhnya untuk mengonfigurasi dan menguji pengaturan tambahan seperti penskalaan kapasitas otomatis.

- Package build server game Anda dan unggah keAmazon GameLift Servers. Buat skrip penginstalan dengan file build, dependensi, dan perangkat lunak pendukung Anda. Anda dapat menggunakan perangkat lunak build yang sama dengan Anywhere dan armada terkelola Anda. Lihat [Menerapkan build server khusus untuk Amazon GameLift Servers mengadakan](#). Anda dapat mengunggah build Anda Amazon GameLift Servers menggunakan konsol atau AWS CLI.

Sebelum mengunggah build Anda, putuskan apa yang ingin Wilayah AWS Anda buat armada terkelola. Anda harus mengunggah build ke Wilayah yang sama. Untuk informasi lebih lanjut tentang memilih lokasi armada, lihat [Lokasi armada](#).

- Buat EC2 armada yang dikelola. Anda dapat menggunakan Amazon GameLift Servers konsol atau AWS CLI untuk membuat armada yang dikelola. Saat Anda membuat armada, Amazon GameLift Servers segera mulai menerapkan build server game Anda untuk hosting. Anda dapat mengonfigurasi banyak aspek armada yang dikelola. Untuk panduan, lihat [Buat EC2 armada Amazon GameLift Servers terkelola](#). Minimal, lakukan hal berikut:
 - Beri nama armada dan tentukan build game yang diunggah untuk diterapkan.
 - Pilih Instans Sesuai Permintaan untuk armada Anda dan pilih jenis instans yang tersedia di lokasi armada Anda. Armada spot adalah pilihan yang berharga tetapi membutuhkan desain dan konfigurasi tambahan.
 - Buat konfigurasi runtime dengan pengaturan serupa seperti yang Anda gunakan dengan armada Anywhere. Minimal, tentukan jalur peluncuran untuk server game Anda yang dapat dieksekusi.
 - Tentukan pengaturan port untuk memungkinkan lalu lintas masuk mengakses server game Anda.
- Tambahkan armada terkelola ke antrean sesi permainan bersama Anda. Perbarui antrian dari Langkah 4 sehingga mencakup tujuan untuk armada terkelola dan armada Anywhere yang dikerahkan dengan Agen. Amazon GameLift Servers
- Uji hosting game dengan armada terkelola Anda. Pada titik ini Anda harus dapat menguji seluruh siklus hosting, dengan klien game yang meminta sesi permainan, mendapatkan info koneksi, dan berhasil terhubung ke sesi permainan.

Langkah 6: Sesuaikan armada Anda

Saat Anda mempersiapkan peluncuran game, Anda harus menyempurnakan solusi hosting Anda. Beberapa keputusan yang perlu dipertimbangkan meliputi:

- Untuk armada Anywhere, otomatisasi proses memulai dan mematikan komputasi sesuai kebutuhan, termasuk menginstal dan menjalankan perangkat lunak server game. Komputasi daur ulang berguna untuk memastikan bahwa mereka diperbarui secara berkala, dan mematikan komputasi dapat menghemat biaya saat tidak diperlukan.
- Jika server game Anda perlu mengkomunikasikan AWS sumber daya lain, siapkan peran IAM untuk mengelola akses. Lihat [Berkomunikasi dengan AWS sumber daya lain dari armada Anda](#).
- Tentukan di mana secara geografis Anda ingin memposisikan server game. Tambahkan lokasi jarak jauh ke armada terkelola Anda. Lihat [Sesuaikan Amazon GameLift Servers EC2 armada yang dikelola](#).
- Untuk armada yang dikelola, pertimbangkan untuk menggunakan armada Spot untuk penghematan biaya. Lihat [Tutorial: Membuat Amazon GameLift Servers antrian dengan Instans Spot](#).
- Optimalkan kinerja armada dengan memilih konfigurasi sumber daya komputasi, lalu konfigurasi instruksi runtime Anda untuk menjalankan jumlah optimal proses server per komputasi. Lakukan ini untuk armada Anywhere dan armada yang dikelola. Lihat [Kelola caranya Amazon GameLift Servers meluncurkan server game](#).
- Bereksperimenlah dengan opsi penempatan sesi game untuk armada terkelola, termasuk menyesuaikan pengaturan prioritas. Lihat [Kustomisasi antrian sesi permainan](#).
- Untuk armada yang dikelola, atur penskalaan kapasitas otomatis untuk memenuhi permintaan pemain yang diharapkan. Lihat [Menskalakan kapasitas hosting game dengan Amazon GameLift Servers](#).
- Untuk armada Anywhere, buat mekanisme untuk menangani penskalaan kapasitas manual atau otomatis untuk memenuhi permintaan pemain yang diharapkan.
- Rancang dan implementasikan failover ke sumber daya lain jika diperlukan. Siapkan armada siaga di armada lain Wilayah AWS dan ubah antrian dan penskalaan otomatis untuk menangani failover jika diperlukan.
- Siapkan alat observabilitas hosting, termasuk analitik dan pencatatan. Lihat [Pemantauan Amazon GameLift Servers](#). Buat grup metrik untuk menggabungkan analitik untuk semua sumber daya hosting Anda.

- Otomatiskan penerapan Anda menggunakan [infrastruktur sebagai kode \(IaC\)](#). Lihat [Mengelola Amazon GameLift Servers sumber daya hosting menggunakan AWS CloudFormation](#).

Amazon GameLift Servers mendukung penggunaan AWS CloudFormation template untuk konfigurasi khusus penerapan apa pun. Anda juga dapat menggunakan AWS Cloud Development Kit (AWS CDK) untuk menentukan Amazon GameLift Servers sumber daya Anda. Untuk informasi selengkapnya tentang ini AWS CDK, lihat [Panduan AWS Cloud Development Kit \(AWS CDK\) Pengembang](#).

Untuk mengelola penyebaran AWS CloudFormation tumpukan Anda, sebaiknya gunakan alat dan layanan integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD) seperti. AWS CodePipeline Alat-alat ini membantu Anda menerapkan secara otomatis atau dengan persetujuan setiap kali Anda membangun biner server game. Dengan CI/CD alat atau layanan, penyebaran sumber daya untuk versi server game baru dapat terlihat seperti ini:

- Bangun dan uji biner server game Anda.
- Unggah biner ke Amazon GameLift Servers.
- Terapkan armada baru dengan build baru.
- Tambahkan armada baru ke antrian sesi game Anda dan hapus armada dengan versi build sebelumnya.
- Saat armada dengan build sebelumnya tidak lagi menghosting sesi game aktif, hapus AWS CloudFormation tumpukan armada tersebut.

Mempersiapkan game untuk Amazon GameLift Servers

Siapkan game multipemain Anda untuk hosting Amazon GameLift Servers. Integrasikan Amazon GameLift Servers hosting fitur ke dalam proyek game Anda dan bangun server game dan server klien Anda. Siapkan lingkungan pengujian yang dihosting untuk mendukung pengembangan dan pengujian game iteratif yang cepat.

Topik

- [Integrasikan game dengan server game khusus](#)
- [Rancang layanan klien game Anda](#)
- [Siapkan untuk pengembangan berulang dengan Amazon GameLift Servers Dimanapun](#)
- [Uji integrasi Anda menggunakan Amazon GameLift Servers Lokal:](#)
- [Menambahkan FlexMatch perijodohan](#)
- [Dapatkan data armada untuk sebuah Amazon GameLift Servers instance](#)
- [Mengintegrasikan game dengan Amazon GameLift ServersRealtime](#)

Integrasikan game dengan server game khusus

Amazon GameLift Servers menyediakan set alat lengkap untuk mempersiapkan game multipemain dan server game khusus Anda untuk dijalankan Amazon GameLift Servers. Amazon GameLift Servers SDKs berisi pustaka yang diperlukan untuk klien game dan server untuk berkomunikasi Amazon GameLift Servers. Untuk informasi lebih lanjut tentang SDKs dan di mana mendapatkannya, lihat[Dapatkan alat Amazon GameLift Servers pengembangan](#).

Topik di bagian ini berisi petunjuk terperinci tentang cara menambahkan Amazon GameLift Servers fungsionalitas untuk klien game dan server game Anda sebelum menerapkan Amazon GameLift Servers.

Topik

- [Interaksi klien/server game dengan Amazon GameLift Servers](#)
- [Integrasikan server game Anda dengan Amazon GameLift Servers](#)
- [Integrasikan klien game Anda denganAmazon GameLift Servers](#)
- [Mesin permainan dan Amazon GameLift Servers](#)

Interaksi klien/server game dengan Amazon GameLift Servers

Komponen dalam Amazon GameLift Servers solusi hosting berinteraksi satu sama lain dengan cara tertentu untuk menjalankan sesi permainan dalam menanggapi permintaan pemain. Topik ini menjelaskan bagaimana komponen berkomunikasi satu sama lain saat server game Anda di-host Amazon GameLift Servers EC2 armada yang dikelola, dikelola sendiri Amazon GameLift Servers Di mana saja armada, atau solusi hibrida.

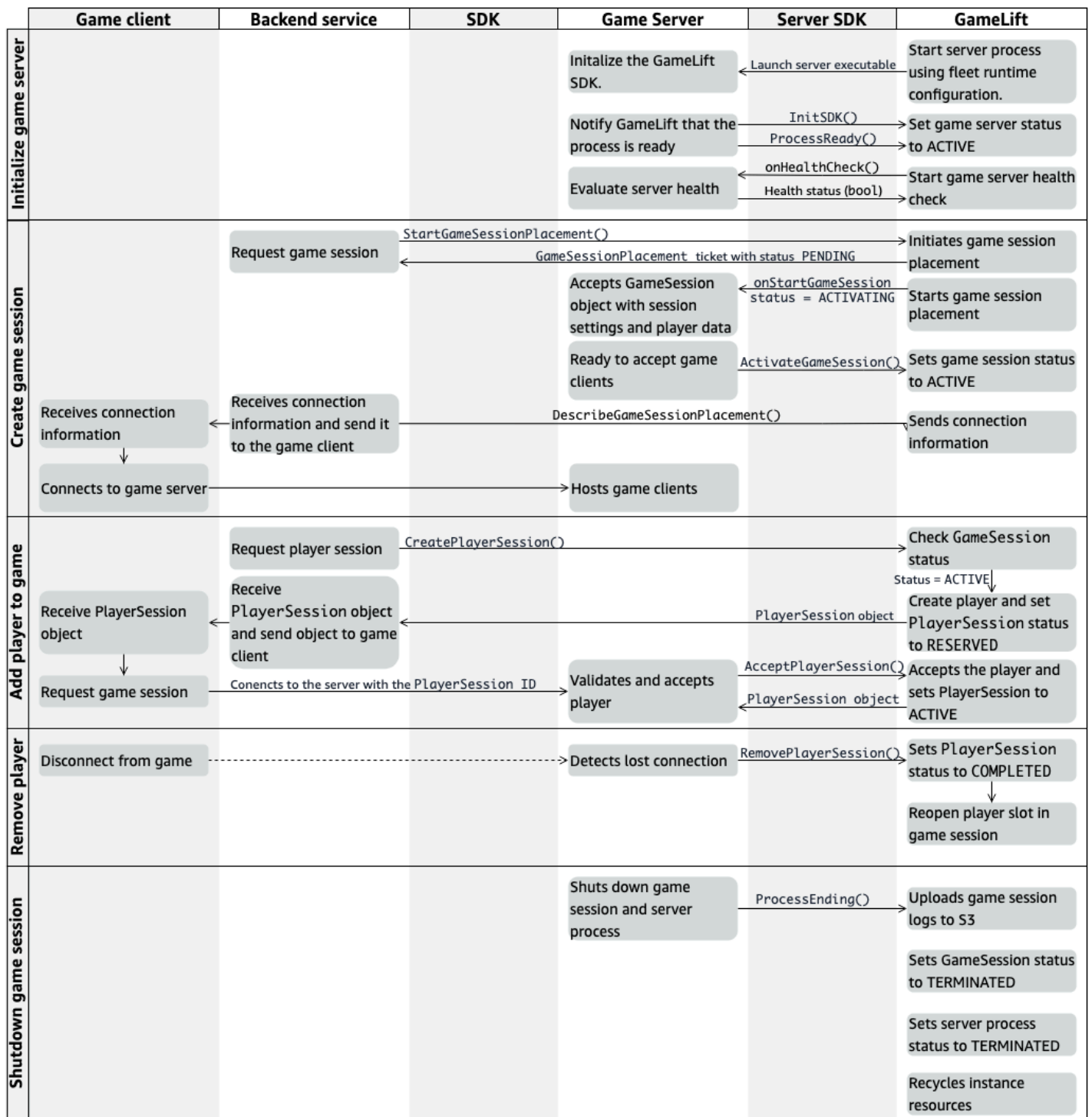
Komponen solusi hosting termasuk server game, Amazon GameLift Servers layanan, layanan backend sisi klien, dan klien game. Server game menggunakan [Amazon GameLift Servers SDK server](#) untuk berinteraksi dengan Amazon GameLift Servers layanan. Layanan backend menggunakan [Amazon GameLift Servers API layanan](#) (bagian dari AWS SDK) untuk berinteraksi dengan layanan atas nama klien game. Saat bergabung dengan sesi permainan, klien game terhubung langsung ke sesi game menggunakan alamat IP dan nomor port unik sesi game.

Topik

- [Diagram interaksi](#)
- [Perilaku interaksi](#)

Diagram interaksi

Diagram berikut menggambarkan bagaimana komponen hosting game Anda berinteraksi sehingga Amazon GameLift Servers Layanan dapat melacak status ketersediaan server game dan memulai sesi permainan dalam menanggapi permintaan pemain.



Perilaku interaksi

Bagian berikut menjelaskan urutan peristiwa di setiap interaksi kunci.

Menginisialisasi proses server game

Saat startup, proses server game membangun komunikasi dengan Amazon GameLift Servers layanan dan melaporkan statusnya sebagai siap untuk menjadi tuan rumah sesi permainan.

1. Proses baru dari server game yang dapat dieksekusi mulai berjalan pada sumber daya hosting.
2. Proses server game memanggil operasi SDK server berikut secara berurutan:
 - a. `InitSDK()` untuk menginisialisasi SDK server, mengautentikasi proses server, dan menjalin komunikasi dengan Amazon GameLift Servers layanan.
 - b. `ProcessReady()` untuk mengkomunikasikan kesiapan untuk menyelenggarakan sesi permainan. Panggilan ini juga melaporkan informasi koneksi proses, yang digunakan klien game untuk terhubung ke sesi permainan, dan informasi lainnya.

Proses server kemudian menunggu permintaan dari Amazon GameLift Servers layanan.

3. Amazon GameLift Servers memperbarui status proses server ke `ACTIVE` dan tersedia untuk meng-host sesi permainan baru.
4. Amazon GameLift Servers dimulai secara berkala memanggil `onHealthCheck` callback untuk meminta status kesehatan dari proses server. Panggilan ini berlanjut saat proses server tetap dalam status aktif. Proses server harus merespon sehat atau tidak sehat dalam satu menit. Jika proses server merespons tidak sehat atau tidak merespons, pada titik tertentu Amazon GameLift Servers Layanan mengubah status aktif proses server dan berhenti mengirim permintaan untuk memulai sesi permainan.

Membuat sesi game

Bagian Amazon GameLift Servers layanan memulai sesi permainan baru dalam menanggapi permintaan dari pemain untuk memainkan game.

1. Seorang pemain yang menggunakan klien game meminta untuk bergabung dengan sesi permainan. Bergantung pada bagaimana game Anda menangani proses bergabung pemain, klien game mengirimkan permintaan ke layanan backend.
2. Jika proses bergabung pemain memerlukan memulai sesi permainan baru, layanan backend mengirimkan permintaan untuk sesi permainan baru ke Amazon GameLift Servers layanan. Permintaan ini memanggil operasi API layanan `StartGameSessionPlacement()`. (Sebagai alternatif, layanan backend mungkin memanggil `StartMatchmaking()`, atau `CreateGameSession()`.)

3. Bagian Amazon GameLift Servers layanan merespons dengan membuat `GameSessionPlacement` tiket baru dengan status `PENDING`. Ini mengembalikan informasi tiket ke layanan backend, sehingga dapat melacak status tiket penempatan dan menentukan kapan sesi permainan siap untuk pemain. Untuk informasi selengkapnya, lihat [Atur notifikasi kejadian untuk penempatan sesi game](#).
4. Bagian Amazon GameLift Servers layanan memulai proses penempatan sesi permainan. Ini mengidentifikasi armada mana yang harus dilihat dan mencari armada tersebut untuk proses server aktif yang tidak menghosting sesi permainan. Saat menemukan proses server yang tersedia, Amazon GameLift Servers layanan melakukan hal berikut:
 - a. Membuat `GameSession` objek dengan pengaturan sesi permainan dan data pemain dari permintaan penempatan, dan menetapkan status ke `ACTIVATING`.
 - b. Meminta proses server untuk memulai sesi permainan. Layanan memanggil `onStartGameSession` callback proses server dan meneruskan objek `GameSession`
 - c. Mengubah jumlah sesi permainan proses server menjadi 1.
5. Proses server menjalankan fungsi `onStartGameSession` callback-nya. Ketika proses server siap menerima koneksi pemain, ia memanggil operasi SDK server `ActivateGameSession()` dan menunggu koneksi pemain.
6. Bagian Amazon GameLift Servers layanan memperbarui `GameSession` objek dengan informasi koneksi untuk proses server (seperti yang dilaporkan dalam panggilan `keProcessReady()`) dan menetapkan status sesi permainan ke `ACTIVE`. Ini juga memperbarui status `GameSessionPlacement` tiket ke `FULFILLED`.
7. Layanan backend memanggil `DescribeGameSessionPlacement()` untuk memeriksa status tiket dan mendapatkan informasi sesi permainan. Saat sesi permainan aktif, layanan backend memberi tahu klien game dan meneruskan informasi koneksi sesi game.
8. Klien game menggunakan informasi koneksi untuk terhubung langsung ke proses server game dan bergabung dengan sesi permainan.

Menambahkan pemain ke game

Sebuah game secara opsional dapat menggunakan sesi pemain untuk melacak koneksi pemain ke sesi permainan. Sesi pemain dapat dibuat secara individual atau sebagai bagian dari permintaan penempatan sesi permainan.

1. Layanan backend memanggil operasi API layanan `CreatePlayerSession()` dengan ID sesi game.

2. Bagian Amazon GameLift Servers Layanan memeriksa status sesi permainan (harus `ACTIVE`), dan mencari slot pemain terbuka di sesi permainan. Jika slot tersedia, maka layanan melakukan hal berikut:
 - a. Membuat `PlayerSession` objek baru dan menetapkan status ke `RESERVED`.
 - b. Menanggapi permintaan layanan backend dengan informasi sesi pemain.
3. Layanan backend meneruskan informasi sesi pemain ke klien game bersama dengan informasi koneksi sesi game.
4. Klien game menggunakan informasi koneksi dan ID sesi pemain untuk terhubung langsung ke proses server game dan meminta untuk bergabung dengan sesi permainan.
5. Menanggapi upaya bergabung klien game, proses server game memanggil operasi API layanan `AcceptPlayerSession()` untuk memvalidasi ID sesi pemain. Proses server bisa menerima atau menolak connection.
6. Bagian Amazon GameLift Servers layanan melakukan salah satu dari yang berikut:
 - a. Jika koneksi diterima, maka Amazon GameLift Servers mengatur `PlayerSession` status ke `ACTIVE` dan meneruskan `PlayerSession` ke proses server game.
 - b. Jika proses server game tidak memanggil `AcceptPlayerSession()` ID sesi pemain dalam jangka waktu tertentu setelah `CreatePlayerSession()` permintaan asli, Amazon GameLift Servers Layanan mengubah `PlayerSession` status `TIMEDOUT` dan membuka kembali slot pemain dalam sesi permainan.

Menghapus pemain

Untuk game yang menggunakan sesi pemain, proses server game memberi tahu Amazon GameLift Servers layanan ketika pemain terputus. Layanan ini menggunakan informasi ini untuk melacak status slot pemain dalam sesi permainan dan dapat memungkinkan pemain baru untuk menggunakan slot terbuka.

1. Seorang pemain terputus dari sesi permainan.
2. Proses server game mendeteksi koneksi yang hilang dan memanggil operasi SDK server `RemovePlayerSession()`
3. Bagian Amazon GameLift Servers Layanan mengubah status sesi pemain `COMPLETED` dan membuka kembali slot pemain dalam sesi permainan.

Mematikan sesi permainan

Pada akhir sesi permainan, atau saat mematikan sesi permainan, proses server memberi tahu Amazon GameLift Servers layanan status sesi permainan.

1. Proses server game mengakhiri sesi permainan dan memulai proses shutdown dengan memanggil operasi SDK server. `ProcessEnding()`
2. Bagian Amazon GameLift Servers layanan melakukan hal berikut:
 - a. Mengunggah log sesi game ke Amazon Simple Storage Service (Amazon S3).
 - b. Mengubah status sesi game menjadi `TERMINATED`.
 - c. Mengubah status proses server menjadi `TERMINATED`.
 - d. Bergantung pada bagaimana solusi hosting dirancang, sumber daya hosting yang baru tersedia dialokasikan untuk menjalankan proses server game baru.

Integrasikan server game Anda dengan Amazon GameLift Servers

Setelah server game kustom Anda diterapkan dan berjalan Amazon GameLift Servers Misalnya, ia harus dapat berinteraksi dengan Amazon GameLift Servers (dan potensi sumber daya lainnya). Bagian ini menjelaskan cara mengintegrasikan perangkat lunak server game Anda dengan Amazon GameLift Servers.

Note

Instruksi ini mengasumsikan bahwa Anda telah membuat Akun AWS dan bahwa Anda memiliki proyek server game yang ada.

Topik di bagian ini mendeskripsikan cara menangani tugas integrasi berikut:

- Membangun komunikasi antara Amazon GameLift Servers dan server game Anda.
- Hasilkan dan gunakan sertifikat TLS untuk membuat koneksi aman antara klien game dan server game.
- Berikan izin untuk perangkat lunak server game Anda untuk berinteraksi dengan AWS sumber daya lain.
- Izinkan proses server game untuk mendapatkan informasi tentang armada yang mereka jalankan.

Topik

- [Tambahkan Amazon GameLift Servers ke server game Anda](#)
- [Berkomunikasi dengan AWS sumber daya lain dari armada Anda](#)

Tambahkan Amazon GameLift Servers ke server game Anda

Topik ini menjelaskan cara memodifikasi kode server game Anda sehingga proses server game dapat berkomunikasi dengan Amazon GameLift Servers layanan. Gunakan petunjuk ini untuk server game yang Anda rencanakan untuk diterapkan ke armada Amazon GameLift Servers terkelola, EC2 armada kontainer terkelola, atau armada Anywhere.

Proses server game berkomunikasi dengan Amazon GameLift Servers layanan untuk menerima instruksi dari layanan dan melaporkan kesehatan proses server dan status sesi game. Untuk informasi rinci tentang interaksi antara komponen solusi hosting game Anda (server game, layanan backend, klien game, dan Amazon GameLift Servers), lihat [Interaksi klien/server game dengan Amazon GameLift Servers](#)

Untuk mempersiapkan game Anda untuk hosting, tambahkan SDK server Amazon GameLift Servers untuk proyek server game Anda. Jika Anda menggunakan Amazon GameLift Servers plugin untuk Unreal Engine atau Unity, SDK server sudah terpasang dan siap digunakan. Server SDK tersedia dalam beberapa bahasa. Untuk informasi selengkapnya tentang dukungan alat untuk server game, termasuk SDK server, lihat [Dapatkan alat Amazon GameLift Servers pengembangan](#).

Referensi API SDK Server:

- [Server C ++ SDK 5.x untuk Amazon GameLift Servers -- Tindakan](#)
- [C# server SDK 5.x untuk Amazon GameLift Servers -- Tindakan](#)
- [Server C ++ \(Tidak nyata\) SDK 5.x untuk -- Tindakan Amazon GameLift Servers](#)
- [Go server SDK untuk Amazon GameLift Servers -- Tindakan](#)

Inisialisasi proses server

Tambahkan kode untuk menjalin komunikasi dengan Amazon GameLift Servers layanan dan laporkan saat proses server game siap menjadi tuan rumah sesi permainan. Kode ini harus berjalan sebelum Amazon GameLift Servers kode apa pun.

1. Inisialisasi klien Amazon GameLift Servers API dengan menelepon `InitSdk()`. [Jika Anda mempersiapkan server game Anda untuk berjalan di EC2 armada Amazon GameLift Servers](#)

[terkelola, gunakan default InitSDK\(\) \(C++\) \(C#\) \(Unreal\) \(Go\) \(C++\) tanpa parameter](#). Klien API menangani koneksi ke Amazon GameLift Servers layanan untuk Anda.

 Jika Anda sedang mempersiapkan server game Anda untuk digunakan pada armada Amazon GameLift Servers Anywhere:

Inisialisasi klien Amazon GameLift Servers API dengan menelepon `InitSdk()` dengan yang berikut: `ServerParameters`

- URL websocket yang digunakan untuk terhubung ke server game Anda.
- ID dari proses yang digunakan untuk meng-host server game Anda.
- ID komputasi yang menghosting proses server game Anda.
- ID armada yang berisi komputasi Amazon GameLift Servers Anywhere Anda.
- Token otorisasi yang dihasilkan oleh Amazon GameLift Servers operasi [GetComputeAuthToken](#).

2. Beri tahu layanan bahwa proses server game siap untuk menjadi tuan rumah sesi permainan. Panggil `ProcessReady()` ([C++](#)) ([C #](#)) ([Unreal](#)) ([Go](#)) ([C++](#)) dengan yang berikut ini. `ProcessParameters` Setiap proses server game harus menelepon `ProcessReady()` hanya sekali.

- Nomor port untuk proses server. Ketika proses server memulai sesi permainan, ia menyediakan port ke Amazon GameLift Servers layanan, yang memperbarui informasi sesi permainan. Game Anda dapat mengambil informasi ini dan memberikannya kepada klien game, yang menggunakannya untuk terhubung ke proses server dan bergabung dengan sesi permainan.
- Lokasi file yang ingin Anda simpan Amazon GameLift Servers untuk Anda. Ini mungkin termasuk log sesi game dan file lain yang dihasilkan oleh proses server selama sesi game. Meskipun Amazon GameLift Servers menyimpan file-file ini sementara pada komputasi di mana proses server berjalan, ini hanya tersedia sampai instance dimatikan. Anda dapat mengakses file yang disimpan melalui [Amazon GameLift Servers konsol](#) atau dengan memanggil operasi Amazon GameLift Servers API [GetGameSessionLogUrl\(\)](#).

- i** Jika Anda sedang mempersiapkan server game untuk digunakan pada armada kontainer Amazon GameLift Servers terkelola:

Anda tidak perlu menentukan parameter log untuk armada kontainer. Sebagai gantinya, kirim sesi game dan data log lainnya ke output standar. Armada kontainer secara otomatis menangkap semua output standar kontainer sebagai aliran log.

- Fungsi callback berikut yang memungkinkan Amazon GameLift Servers untuk mengirim pesan atau prompt ke proses server game. Anda harus menerapkan masing-masing fungsi ini dalam kode server game Anda. [Untuk informasi lebih lanjut, lihat `ProcessParameters\(C++\)` \(`C#`\) \(`Unreal`\) \(`Go`\) \(`C++`\) `Unreal`\).](#)
- (Opsional) `onHealthCheck` — Amazon GameLift Servers memanggil fungsi ini secara teratur untuk meminta laporan status kesehatan dari server.
- `onStartGameSession`— Amazon GameLift Servers memanggil fungsi ini sebagai respons terhadap permintaan klien [CreateGameSession\(\)](#).
- `onProcessTerminate`— Amazon GameLift Servers memaksa proses server berhenti, membiarkannya dimatikan dengan anggun.
- (Opsional) `onUpdateGameSession` — Amazon GameLift Servers mengirimkan objek sesi game yang diperbarui ke server game atau memberikan pembaruan status pada permintaan pengisian ulang pertandingan. Fitur [FlexMatchbackfill](#) membutuhkan callback ini.

Anda juga dapat mengatur server game sehingga dapat mengakses AWS sumber daya lain yang Anda miliki atau kendalikan dengan aman. Untuk informasi selengkapnya, lihat [Berkomunikasi dengan AWS sumber daya lain dari armada Anda](#).

(Opsional) Laporkan kesehatan proses server

Tambahkan kode ke server game Anda untuk mengimplementasikan fungsi `onHealthCheck()` callback. Amazon GameLift Servers memanggil metode callback ini secara berkala untuk mengumpulkan metrik kesehatan. Untuk menerapkan fungsionalitas callback ini, lakukan hal berikut:

- Mengevaluasi status kesehatan dari proses server. Misalnya, Anda mungkin melaporkan proses server sebagai tidak sehat jika ada dependensi eksternal yang gagal.

- Selesaikan evaluasi kondisi dan tanggap panggilan balik dalam waktu 60 detik. Jika Amazon GameLift Servers tidak menerima respons pada waktu itu, secara otomatis menganggap proses server tidak sehat.
- Kembalikan nilai Boolean: true untuk sehat, false untuk tidak sehat.

Jika Anda tidak menerapkan callback pemeriksaan kesehatan, maka Amazon GameLift Servers menganggap proses server menjadi sehat kecuali server tidak merespons.

Amazon GameLift Servers Layanan ini menggunakan kesehatan proses server untuk mengakhiri proses yang tidak sehat dan membersihkan sumber daya. Jika proses server terus melaporkan sebagai tidak sehat atau tidak merespons untuk tiga pemeriksaan kesehatan berturut-turut, maka layanan mungkin mematikan proses dan memulai yang baru. Layanan ini mengumpulkan metrik pada kesehatan proses server armada.

(Opsional) Dapatkan sertifikat TLS

Jika proses server berjalan pada armada yang mengaktifkan pembuatan sertifikat TLS, maka Anda dapat mengambil sertifikat TLS untuk membuat koneksi aman dengan klien game dan untuk mengenkripsi komunikasi server klien. Salinan sertifikat disimpan pada instansi. [Untuk mendapatkan lokasi file, panggil `GetComputeCertificate\(\)` \(C++\) \(C #\) \(Unreal\) \(Go\) \(C++\)Unreal\).](#)

Mulai sesi game

Tambahkan kode untuk mengimplementasikan fungsi `onStartGameSession` callback. Amazon GameLift Servers memanggil callback ini untuk memulai sesi permainan pada proses server.

`onStartGameSession` Fungsi mengambil [GameSession](#) objek sebagai parameter input. Objek ini mencakup informasi sesi permainan utama, seperti pemain maksimum. Ini juga dapat mencakup data game dan data pemain. Implementasi fungsi harus melakukan tugas-tugas berikut:

- Lakukan tindakan untuk membuat sesi game baru berdasarkan `GameSession` properti. Minimal, server game harus mengaitkan ID sesi game, yang dirujuk klien game saat menghubungkan ke proses server.
- Data proses game dan data pemain yang diperlukan. Data ini ada di `GameSession` objek.
- Beri tahu Amazon GameLift Servers layanan saat sesi permainan baru siap menerima pemain. [Panggil operasi API server `ActivateGameSession\(\)` \(C++\) \(C #\) \(Unreal\) \(Go\) \(C++\)Tidak nyata\).](#) Menanggapi panggilan yang berhasil, layanan mengubah status sesi game menjadi `ACTIVE`.

(Opsional) Validasi pemain baru

Jika Anda melacak status sesi pemain, tambahkan kode untuk memvalidasi pemain baru saat mereka terhubung ke server game. Amazon GameLift Servers melacak pemain saat ini dan slot sesi permainan yang tersedia.

Untuk validasi, klien game yang mencoba bergabung dengan sesi permainan harus menyertakan ID sesi pemain. Amazon GameLift Servers menghasilkan ID ini saat game Anda memulai sesi permainan baru dengan memanggil [StartGameSessionPlacement\(\)](#) atau [StartMatchmaking\(\)](#). Atas permintaan ini, slot terbuka dalam sesi permainan dicadangkan untuk sesi pemain.

Ketika proses server game menerima permintaan koneksi klien game, ia memanggil `AcceptPlayerSession()` ([C++](#)) ([C#](#)) ([Unreal](#)) ([Go](#)) ([C++](#)) dengan ID sesi pemain. Sebagai tanggapan, Amazon GameLift Servers verifikasi bahwa ID sesi pemain sesuai dengan slot terbuka yang dicadangkan dalam sesi permainan. Setelah Amazon GameLift Servers memvalidasi ID sesi pemain, proses server menerima koneksi. Pemain kemudian dapat bergabung dengan sesi permainan. Jika Amazon GameLift Servers tidak memvalidasi ID sesi pemain, maka proses server menyangkal koneksi.

(Opsional) Laporkan akhir sesi pemain

Jika Anda melacak status sesi pemain, tambahkan kode untuk memberi tahu Amazon GameLift Servers saat pemain meninggalkan sesi permainan. Kode ini harus berjalan setiap kali proses server mendeteksi koneksi terputus. Amazon GameLift Servers menggunakan pemberitahuan ini untuk melacak pemain saat ini dan slot yang tersedia di sesi permainan.

Untuk menangani koneksi terputus dalam kode Anda, tambahkan panggilan ke operasi API server `RemovePlayerSession()` ([C++](#)) ([C#](#)) ([Unreal](#)) ([Go](#)) ([C++](#)) dengan ID sesi pemain yang sesuai.

Mengakhiri sesi game

Tambahkan kode ke urutan shutdown proses server untuk memberi tahu Amazon GameLift Servers saat sesi permainan berakhir. Untuk mendaur ulang dan menyegarkan sumber daya hosting, matikan setiap proses server setelah sesi permainan selesai.

Pada awal kode shutdown proses server, panggil operasi API server ([C++](#)) `ProcessEnding()` ([C#](#)) ([Unreal](#)) ([Go](#)) g ([C++](#)) (. Panggilan ini memberi tahu Amazon GameLift Servers bahwa proses server dimatikan. Amazon GameLift Servers mengubah status sesi game dan status proses server menjadi `TERMINATED`. Setelah menelepon `ProcessEnding()`, aman untuk proses ditutup.

Menanggapi notifikasi shutdown proses server

Tambahkan kode untuk mematikan proses server sebagai tanggapan atas pemberitahuan dari Amazon GameLift Servers layanan. Layanan mengirimkan pemberitahuan ini ketika proses server secara konsisten melaporkan tidak sehat, atau jika instance di mana proses server berjalan sedang dihentikan. Amazon GameLift Servers dapat menghentikan instance sebagai bagian dari peristiwa penurunan skala kapasitas, atau sebagai respons terhadap gangguan Instans Spot. Interupsi Instans Spot memberikan pemberitahuan dua menit, yang memberi waktu proses server untuk memutuskan sambungan pemain dengan anggun, menyimpan data status game, dan melakukan tugas pembersihan lainnya.

Untuk menangani notifikasi shutdown, buat perubahan berikut pada kode server game Anda:

- [Menerapkan fungsi callback `onProcessTerminate\(\)` \(C++\) \(C#\) \(Unreal\) \(Go\) \(C++\)](#). Fungsi ini harus memanggil kode yang mematikan proses server.
- Panggil operasi API server `GetTerminationTime()` (C++) (C#) (Unreal) (Go) (C++) server game Anda. Jika Amazon GameLift Servers telah mengeluarkan panggilan untuk menghentikan proses server, maka `GetTerminationTime()` mengembalikan perkiraan waktu penghentian.
- [Pada awal kode shutdown server game Anda, panggil operasi API server \(C++\) `ProcessEnding\(\)` \(C#\) \(Unreal\) \(Go\) \(C++\) \(C#\)](#). Panggilan ini memberi tahu Amazon GameLift Servers layanan bahwa proses server dimatikan. Layanan kemudian mengubah status proses server menjadi `TERMINATED`. Setelah menelepon `ProcessEnding()`, aman untuk proses ditutup.

Berkomunikasi dengan AWS sumber daya lain dari armada Anda

Saat membuat build server game untuk penyebaran di Amazon GameLift Servers armada, Anda mungkin ingin aplikasi di build game Anda berkomunikasi secara langsung dan aman dengan AWS sumber daya lain yang Anda miliki. Karena Amazon GameLift Servers mengelola armada hosting game Anda, Anda harus memberikan akses Amazon GameLift Servers terbatas ke sumber daya dan layanan ini.

Beberapa contoh skenario meliputi:

- Gunakan CloudWatch agen Amazon untuk mengumpulkan metrik, log, dan jejak dari armada terkelola dan EC2 armada Anywhere.
- Kirim data log instance ke Amazon CloudWatch Logs.

- Dapatkan file game yang disimpan dalam bucket Amazon Simple Storage Service (Amazon S3).
- Membaca dan menulis data game (seperti mode permainan atau inventaris) yang disimpan dalam database Amazon DynamoDB atau layanan penyimpanan data lainnya.
- Kirim sinyal langsung ke instance menggunakan Amazon Simple Queue Service (Amazon SQS).
- Akses sumber daya khusus yang digunakan dan dijalankan di Amazon Elastic Compute Cloud (Amazon EC2).

Amazon GameLift Servers mendukung metode ini untuk membangun akses:

- [Akses AWS sumber daya dengan peran IAM](#)
- [Akses AWS sumber daya dengan VPC peering](#)

Akses AWS sumber daya dengan peran IAM

Gunakan peran IAM untuk menentukan siapa yang dapat mengakses sumber daya Anda dan menetapkan batasan pada akses tersebut. Pihak tepercaya dapat “mengambil” peran dan mendapatkan kredensial keamanan sementara yang memberi wewenang kepada mereka untuk berinteraksi dengan sumber daya. Ketika para pihak membuat permintaan API yang terkait dengan sumber daya, mereka harus menyertakan kredensialnya.

Untuk mengatur akses yang dikendalikan oleh peran IAM, lakukan tugas-tugas berikut:

1. [Buat peran IAM](#)
2. [Memodifikasi aplikasi untuk memperoleh kredensial](#)
3. [Kaitkan armada dengan peran IAM](#)

Buat peran IAM

Pada langkah ini, Anda membuat peran IAM, dengan serangkaian izin untuk mengontrol akses ke AWS sumber daya Anda dan kebijakan kepercayaan yang memberikan Amazon GameLift Servers hak untuk menggunakan izin peran tersebut.

Untuk petunjuk tentang cara mengatur peran IAM, lihat [Menyiapkan peran layanan IAM untuk Amazon GameLift Servers](#). Saat membuat kebijakan izin, pilih layanan, sumber daya, dan tindakan tertentu yang perlu dikerjakan oleh aplikasi Anda. Sebagai praktik terbaik, batasi ruang lingkup izin sebanyak mungkin.

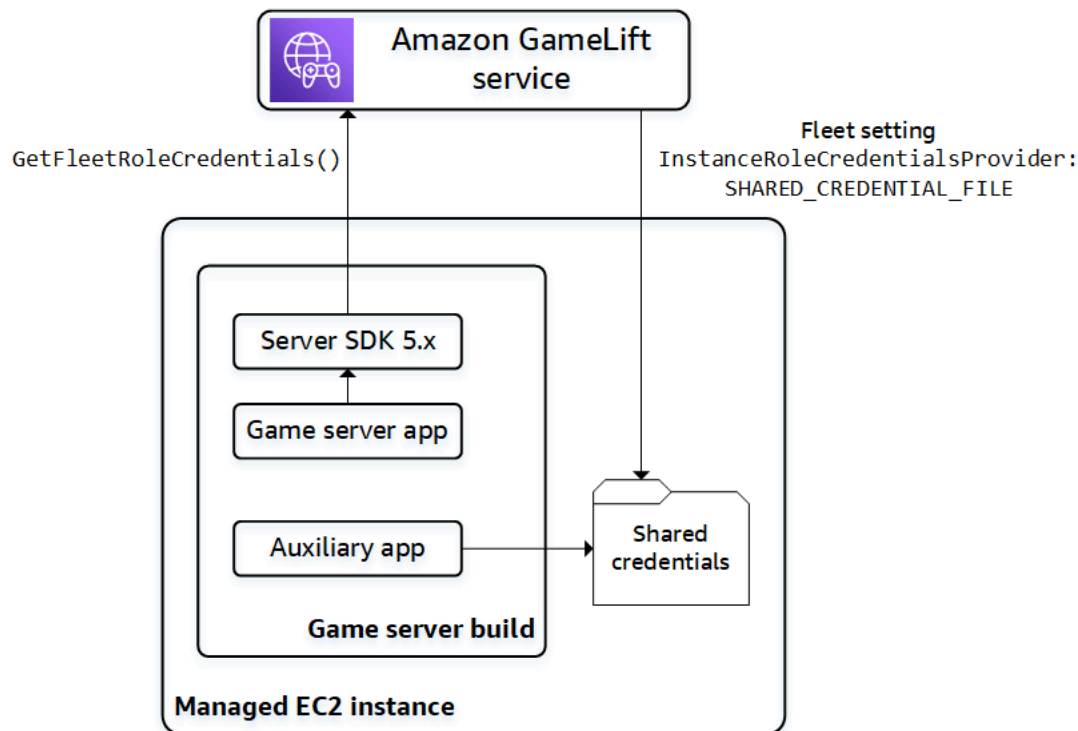
Setelah Anda membuat peran, perhatikan nama sumber daya Amazon (ARN) peran tersebut. Anda membutuhkan peran ARN selama pembuatan armada.

Memodifikasi aplikasi untuk memperoleh kredensial

Pada langkah ini, Anda mengonfigurasi aplikasi Anda untuk memperoleh kredensial keamanan untuk peran IAM dan menggunakannya saat berinteraksi dengan sumber daya Anda. AWS Lihat tabel berikut untuk menentukan cara memodifikasi aplikasi Anda berdasarkan (1) jenis aplikasi, dan (2) versi SDK server yang digunakan game Anda untuk berkomunikasi Amazon GameLift Servers.

	Aplikasi server game	Aplikasi lainnya
Menggunakan server SDK versi 5.x	Panggil metode SDK server <code>GetFleetRoleCredentials()</code> dari kode server game Anda.	Tambahkan kode ke aplikasi untuk menarik kredensial dari file bersama pada instance armada.
Menggunakan server SDK versi 4 atau yang lebih lama	Panggil AWS Security Token Service (AWS STS) AssumeRole dengan peran ARN.	Panggil AWS Security Token Service (AWS STS) AssumeRole dengan peran ARN.

Untuk game yang terintegrasi dengan server SDK 5.x, diagram ini menggambarkan bagaimana aplikasi dalam build game yang Anda gunakan dapat memperoleh kredensial untuk peran IAM.



Panggilan **GetFleetRoleCredentials()** (server SDK 5.x)

Dalam kode server game Anda, yang seharusnya sudah terintegrasi dengan Amazon GameLift Servers server SDK 5.x, panggil `GetFleetRoleCredentials` ([C++](#)) ([C#](#)) ([Unreal](#)) ([Go](#)) untuk mengambil satu set kredensi sementara. Saat kredensialnya kedaluwarsa, Anda dapat menyegarkannya dengan panggilan lain. `GetFleetRoleCredentials`

Gunakan kredensial bersama (server SDK 5.x)

Untuk aplikasi non-server yang digunakan dengan build server game menggunakan server SDK 5.x, tambahkan kode untuk mendapatkan dan menggunakan kredensial yang disimpan dalam file bersama. Amazon GameLift Servers menghasilkan profil kredensial untuk setiap instance armada. Kredensialnya tersedia untuk digunakan oleh semua aplikasi pada instance. Amazon GameLift Server terus menyegarkan kredensial sementara.

Anda harus mengonfigurasi armada untuk menghasilkan file kredensial bersama pada pembuatan armada.

Di setiap aplikasi yang perlu menggunakan file kredensial bersama, tentukan lokasi file dan nama profil, sebagai berikut:

Windows:

```
[credentials]
shared_credential_profile= "FleetRoleCredentials"
shared_credential_file= "C:\\Credentials\\credentials"
```

Linux:

```
[credentials]
shared_credential_profile= "FleetRoleCredentials"
shared_credential_file= "/local/credentials/credentials"
```

Contoh: Siapkan CloudWatch agen untuk mengumpulkan metrik untuk instance Amazon GameLift Servers armada

Jika Anda ingin menggunakan CloudWatch agen Amazon untuk mengumpulkan metrik, log, dan jejak dari Amazon GameLift Servers armada Anda, gunakan metode ini untuk mengotorisasi agen agar memancarkan data ke akun Anda. Dalam skenario ini, ambil langkah-langkah berikut:

1. Ambil atau tulis `config.json` file CloudWatch agen.
2. Perbarui `common-config.toml` file untuk agen untuk mengidentifikasi nama file kredensial dan nama profil, seperti dijelaskan di atas.
3. Siapkan skrip instalasi build server game Anda untuk menginstal dan memulai CloudWatch agen.

Gunakan **`AssumeRole()`** (server SDK 4)

Tambahkan kode ke aplikasi Anda untuk mengambil peran IAM dan mendapatkan kredensial untuk berinteraksi dengan sumber daya Anda. AWS Setiap aplikasi yang berjalan pada instance Amazon GameLift Servers armada dengan server SDK 4 atau sebelumnya dapat mengambil peran IAM.

Dalam kode aplikasi, sebelum mengakses AWS sumber daya, aplikasi harus memanggil operasi [AssumeRole](#) API AWS Security Token Service (AWS STS) dan menentukan peran ARN. Operasi ini mengembalikan satu set kredensi sementara yang mengotorisasi aplikasi untuk mengakses sumber daya. AWS Untuk informasi selengkapnya, lihat [Menggunakan kredensial sementara dengan AWS sumber daya](#) di Panduan Pengguna IAM.

Kaitkan armada dengan peran IAM

Setelah Anda membuat peran IAM dan memperbarui aplikasi di build server game Anda untuk mendapatkan dan menggunakan kredensial akses, Anda dapat menerapkan armada. Saat Anda mengonfigurasi armada baru, atur parameter berikut:

- [InstanceRoleArn](#)— Atur parameter ini ke ARN dari peran IAM.
- [InstanceRoleCredentialsProvider](#)— Untuk meminta Amazon GameLift Servers untuk menghasilkan file kredensi bersama untuk setiap instance armada, setel parameter ini ke. `SHARED_CREDENTIAL_FILE`

Anda harus menetapkan nilai-nilai ini ketika Anda membuat armada. Mereka tidak dapat diperbarui nanti.

Akses AWS sumber daya dengan VPC peering

Anda dapat menggunakan peering Amazon Virtual Private Cloud (Amazon VPC) untuk berkomunikasi antara aplikasi yang berjalan pada Amazon GameLift Servers instans dan sumber daya lain. AWS VPC adalah jaringan pribadi virtual yang Anda tentukan yang mencakup serangkaian sumber daya yang dikelola melalui Anda. Akun AWS Setiap Amazon GameLift Servers armada memiliki VPC sendiri. Dengan VPC peering, Anda dapat membuat koneksi jaringan langsung antara VPC untuk armada Anda dan sumber daya lainnya. AWS

Amazon GameLift Serversmerampingkan proses pengaturan koneksi peering VPC untuk server game Anda. Ini menangani permintaan peering, update tabel rute, dan mengkonfigurasi connection yang diperlukan. Untuk petunjuk tentang cara mengatur peering VPC untuk server game Anda, lihat. [VPC mengintip untuk Amazon GameLift Servers](#)

Integrasikan klien game Anda denganAmazon GameLift Servers

Topik di bagian ini menjelaskan yang dikelola Amazon GameLift Servers fungsionalitas yang dapat Anda tambahkan ke layanan backend. Layanan backend menangani tugas-tugas berikut:

- Meminta informasi tentang sesi permainan aktif dari Amazon GameLift Servers.
- Bergabung dengan pemain ke sesi permainan yang ada.
- Membuat sesi permainan baru dan bergabung dengan pemain untuk itu.
- Mengubah metadata untuk sesi game yang ada.

Untuk informasi lebih lanjut tentang bagaimana klien game berinteraksi Amazon GameLift Servers dan server game berjalan di Amazon GameLift Servers, lihat [Interaksi klien/server game dengan Amazon GameLift Servers](#).

Prasyarat

- Sebuah Akun AWS.
- Build server game yang diunggah ke Amazon GameLift Servers.
- Armada untuk hosting game Anda.

Topik

- [Tambahkan Amazon GameLift Servers ke klien game Anda](#)
- [Hasilkan pemain IDs](#)

Tambahkan Amazon GameLift Servers ke klien game Anda

Integrasikan Amazon GameLift Servers ke dalam komponen game yang membutuhkan informasi sesi game, buat sesi game baru, dan tambahkan pemain ke game. Bergantung pada arsitektur game Anda, fungsi ini ada di layanan backend yang menangani tugas-tugas seperti otentikasi pemain, perjodohan, atau penempatan sesi game.

Note

Untuk informasi mendetail tentang cara mengatur perjodohan untuk game Anda, lihat Panduan [Amazon GameLift ServersFlexMatchPengembang](#).

Siapkan Amazon GameLift Servers pada layanan backend

Tambahkan kode untuk menginisialisasi Amazon GameLift Servers klien dan menyimpan pengaturan kunci. Kode ini harus berjalan sebelum kode apa pun bergantung pada Amazon GameLift Servers.

1. Siapkan konfigurasi klien. Gunakan konfigurasi klien default atau buat objek konfigurasi klien kustom. Untuk informasi selengkapnya, lihat [AWS::Client::ClientConfiguration](#)(C++) atau [AmazonGameLiftConfig](#)(C#).

Konfigurasi klien menentukan wilayah target dan titik akhir yang akan digunakan saat menghubungi. Amazon GameLift Servers Wilayah mengidentifikasi kumpulan sumber daya yang digunakan (armada, antrian, dan mak comblang) untuk digunakan. Konfigurasi klien default menetapkan lokasi ke Wilayah AS Timur (Virginia N.). Untuk menggunakan Wilayah lain, buat konfigurasi khusus.

2. Inisialisasi Amazon GameLift Servers klien. Gunakan [Aws:GameLift::: GameLiftClient \(\)](#) (C + +) atau [AmazonGameLiftClient\(\)](#) (C #) dengan konfigurasi klien default atau konfigurasi klien khusus.
3. Tambahkan mekanisme untuk menghasilkan pengenalan unik bagi setiap pemain. Untuk informasi selengkapnya, lihat [Hasilkan pemain IDs](#).
4. Kumpulkan dan simpan informasi berikut:
 - Armada target - Banyak permintaan Amazon GameLift Servers API harus menentukan armada. Untuk melakukannya, gunakan ID armada atau ID alias yang menunjuk ke armada target. Sebagai praktik terbaik, gunakan alias armada sehingga Anda dapat mengalihkan pemain dari satu armada ke armada lainnya tanpa harus memperbarui layanan backend Anda.
 - Antrian target — Untuk game yang menggunakan antrian multi-armada untuk menempatkan sesi permainan baru, tentukan nama antrian yang akan digunakan.
 - AWS kredensial — Semua panggilan Amazon GameLift Servers harus memberikan kredensial untuk Akun AWS yang menjadi tuan rumah game. Anda memperoleh kredensi ini dengan membuat pengguna pemain, seperti yang dijelaskan dalam [Siapkan akses terprogram untuk game Anda](#) Bergantung pada bagaimana Anda mengelola akses untuk pengguna pemain, lakukan hal berikut:
 - Jika Anda menggunakan peran untuk mengelola izin pengguna pemain, tambahkan kode untuk mengambil peran sebelum memanggil Amazon GameLift Servers API. Permintaan untuk mengambil peran mengembalikan satu set kredensial keamanan sementara. Untuk informasi selengkapnya, lihat [Beralih ke peran IAM \(AWS API\)](#) di Panduan Pengguna IAM.
 - Jika Anda memiliki kredensial keamanan jangka panjang, konfigurasikan kode Anda untuk menemukan dan menggunakan kredensial yang disimpan. Lihat [Mengautentikasi menggunakan kredensial jangka panjang](#) di dalam Panduan Referensi Alat AWS SDKs dan Alat. Untuk informasi tentang menyimpan kredensial, lihat referensi AWS API untuk [\(C++\)](#) dan [\(.NET\)](#).
 - Jika Anda memiliki kredensial keamanan sementara, tambahkan kode untuk menyegarkan kredensial secara teratur menggunakan AWS Security Token Service (AWS STS), seperti yang dijelaskan dalam [Menggunakan kredensial keamanan sementara dengan](#) Panduan Pengguna IAM. AWS SDKs Kode harus meminta kredensi baru sebelum yang lama kedaluwarsa.

Dapatkan sesi permainan

Tambahkan kode untuk menemukan sesi permainan yang tersedia dan mengelola pengaturan sesi permainan dan metadata.

Cari sesi permainan aktif

Gunakan [SearchGameSessions](#) untuk mendapatkan informasi tentang sesi permainan tertentu, semua sesi aktif, atau sesi yang memenuhi serangkaian kriteria pencarian. Panggilan ini mengembalikan [GameSession](#) objek untuk setiap sesi permainan aktif yang cocok dengan permintaan pencarian Anda.

Gunakan kriteria pencarian untuk mendapatkan daftar sesi game aktif yang telah difilter agar pemain dapat bergabung. Misalnya, Anda dapat memfilter sesi sebagai berikut:

- Kecualikan sesi permainan yang penuh: `CurrentPlayerSessionCount = MaximumPlayerSessionCount`.
- Pilih sesi permainan berdasarkan lamanya waktu sesi telah berjalan: `EvaluasiCreationTime`.
- Temukan sesi game berdasarkan properti game khusus: `gameSessionProperties.gameMode = "brawl"`.

Kelola sesi permainan

Gunakan salah satu operasi berikut untuk mengambil atau memperbarui informasi sesi game.

- [DescribeGameSessionDetails\(\)](#) — Dapatkan status perlindungan sesi game selain informasi sesi game.
- [UpdateGameSession\(\)](#) — Ubah metadata dan pengaturan sesi game sesuai kebutuhan.
- [GetGameSessionLogUrl](#) — Akses log sesi game yang disimpan.

Buat sesi permainan

Tambahkan kode untuk memulai sesi game baru di armada yang Anda gunakan dan sediakan untuk pemain. Ada dua opsi untuk membuat sesi permainan, tergantung pada apakah Anda menerapkan game Anda di beberapa Wilayah AWS atau dalam satu Wilayah.

Buat sesi permainan dalam antrian multi-lokasi

Gunakan [StartGameSessionPlacement](#) untuk menempatkan permintaan untuk sesi permainan baru dalam antrian. Untuk menggunakan operasi ini, buat antrian. Ini menentukan di mana Amazon GameLift Servers menempatkan sesi permainan baru. Untuk informasi lebih lanjut tentang antrian dan cara menggunakannya, lihat. [Mengelola penempatan sesi permainan dengan Amazon GameLift Servers mengantri](#)

Saat membuat penempatan sesi permainan, tentukan nama antrian yang akan digunakan, nama sesi game, jumlah maksimum pemain bersamaan, dan kumpulan properti game opsional. Anda juga dapat secara opsional memberikan daftar pemain untuk secara otomatis bergabung dengan sesi permainan. Jika Anda menyertakan data latensi pemain untuk Wilayah yang relevan, Amazon GameLift Servers gunakan informasi ini untuk menempatkan sesi permainan baru pada armada yang memberikan pengalaman gameplay yang ideal bagi para pemain.

Untuk mendapatkan pengukuran latensi yang akurat, gunakan ping Amazon GameLift Servers beacon UDP. Titik akhir ini memungkinkan Anda mengukur latensi jaringan UDP aktual antara perangkat pemain dan lokasi hosting potensial, menghasilkan keputusan penempatan yang lebih akurat daripada menggunakan ping ICMP. Untuk informasi lebih lanjut tentang penggunaan suar ping UDP untuk mengukur latensi, lihat. [Suar ping UDP](#)

Penempatan sesi game adalah proses asynchronous. Setelah Anda mengajukan permintaan, Anda dapat membiarkannya berhasil atau time out. Anda juga dapat membatalkan permintaan kapan saja menggunakan [StopGameSessionPlacement](#). Untuk memeriksa status permintaan penempatan Anda, hubungi [DescribeGameSessionPlacement](#).

Buat sesi permainan pada armada tertentu

Gunakan [CreateGameSession](#) untuk membuat sesi baru pada armada tertentu. Keberhasilan operasi sinkron ini tergantung pada apakah armada memiliki sumber daya yang tersedia untuk menjadi host sesi game baru. Setelah Amazon GameLift Servers membuat sesi permainan baru dan mengembalikan [GameSession](#) objek, Anda dapat bergabung dengan pemain untuk itu.

Saat Anda menggunakan operasi ini, berikan ID armada atau ID alias, nama sesi, dan jumlah maksimum pemain bersamaan untuk permainan. Atau, Anda dapat menyertakan satu set properti game. Properti permainan didefinisikan dalam array pasangan kunci-nilai.

Jika Anda menggunakan fitur perlindungan Amazon GameLift Servers sumber daya untuk membatasi jumlah sesi permainan yang dapat dibuat oleh satu pemain, berikan ID pemain pembuat sesi game.

Bergabunglah dengan pemain ke sesi permainan

Tambahkan kode untuk memesan slot pemain dalam sesi permainan aktif dan menghubungkan klien game ke sesi permainan.

1. Pesan slot pemain dalam sesi permainan

Untuk menyimpan slot pemain, buat sesi pemain baru untuk sesi game. Untuk informasi selengkapnya tentang sesi pemain, lihat [Bagaimana pemain terhubung ke game](#).

Ada dua cara untuk membuat sesi pemain baru:

- Gunakan [StartGameSessionPlacement](#) untuk memesan slot untuk satu atau lebih pemain di sesi permainan baru.
- Cadangan slot pemain untuk satu atau lebih pemain yang menggunakan [CreatePlayerSession](#) atau [CreatePlayerSessions](#) dengan ID sesi permainan.

Amazon GameLift Servers pertama memverifikasi bahwa sesi permainan menerima pemain baru dan memiliki slot pemain yang tersedia. Jika berhasil Amazon GameLift Servers, pesan slot untuk pemain, buat sesi pemain baru, dan kembalikan [PlayerSession](#) objek. Objek ini berisi nama DNS, alamat IP, dan port yang dibutuhkan klien game untuk terhubung ke sesi permainan.

Permintaan sesi pemain harus menyertakan ID unik untuk setiap pemain. Untuk informasi selengkapnya, lihat [Hasilkan pemain IDs](#).

Sesi pemain dapat mencakup satu set data pemain khusus. Data ini disimpan dalam objek sesi pemain yang baru dibuat, yang dapat Anda ambil dengan memanggil [DescribePlayerSessions\(\)](#). Amazon GameLift Servers juga meneruskan objek ini ke server game ketika pemain terhubung langsung ke sesi permainan. Saat meminta beberapa sesi pemain, berikan serangkaian data pemain untuk setiap pemain yang dipetakan ke ID pemain dalam permintaan.

2. Connect ke sesi permainan

Menambahkan kode untuk klien game guna mengambil objek `PlayerSession`, yang berisi informasi connection sesi game. Gunakan informasi ini untuk membuat koneksi langsung ke server.

- Anda dapat terhubung menggunakan port yang ditentukan dan nama DNS atau alamat IP yang ditetapkan untuk proses server.

- Jika armada Anda mengaktifkan pembuatan sertifikat TLS, sambungkan menggunakan nama dan port DNS.
- Jika server game Anda memvalidasi koneksi pemain yang masuk, maka rujuk ID sesi pemain.

Setelah melakukan koneksi, klien game dan proses server berkomunikasi secara langsung tanpa melibatkan Amazon GameLift Servers. Server memelihara komunikasi dengan Amazon GameLift Servers untuk melaporkan status koneksi pemain, status kesehatan, dan banyak lagi. Jika server game memvalidasi pemain yang masuk, maka itu memverifikasi bahwa ID sesi pemain cocok dengan slot yang dicadangkan dalam sesi permainan, dan menerima atau menolak koneksi pemain. Saat pemain terputus, proses server melaporkan connection yang terhenti.

Gunakan properti sesi game

Klien game Anda dapat meneruskan data ke sesi permainan dengan menggunakan properti game. Properti game adalah pasangan nilai kunci yang dapat ditambahkan, dibaca, dicantumkan, dan diubah oleh server game Anda. Anda dapat meneruskan properti game saat membuat sesi permainan baru, atau nanti saat sesi permainan aktif. Sesi permainan dapat berisi hingga 16 properti game. Anda tidak dapat menghapus properti game.

Misalnya, game Anda menawarkan tingkat kesulitan ini: `Novice`, `Easy`, `Intermediate`, dan `Expert`. Seorang pemain memilih `Easy`, dan kemudian memulai permainan. Klien game Anda meminta sesi permainan baru Amazon GameLift Servers dengan menggunakan salah satu `StartGameSessionPlacement` atau `CreateGameSession` seperti yang dijelaskan di bagian sebelumnya. Dalam permintaan tersebut, klien melewati ini: `{"Key": "Difficulty", "Value": "Easy"}`.

Menanggapi permintaan, Amazon GameLift Servers membuat `GameSession` objek yang berisi properti game tertentu. Amazon GameLift Servers kemudian menginstruksikan server game yang tersedia untuk memulai sesi permainan baru dan melewati `GameSession` objek. Server game memulai sesi permainan dengan `Difficulty aEasy`.

Pelajari selengkapnya

- [GameProperty tipe data](#)
- [SearchGameSessions\(\) contoh](#)
- [UpdateGameSession\(\) GameProperties parameter](#)

Hasilkan pemain IDs

Amazon GameLift Servers menggunakan sesi pemain untuk mewakili pemain yang terhubung ke sesi permainan. Amazon GameLift Servers membuat sesi pemain setiap kali pemain terhubung ke sesi permainan menggunakan klien game yang terintegrasi dengan Amazon GameLift Servers. Ketika seorang pemain meninggalkan permainan, sesi pemain berakhir. Amazon GameLift Servers tidak menggunakan kembali sesi pemain.

Contoh kode berikut secara acak menghasilkan pemain IDs unik:

```
bool includeBrackets = false;
bool includeDashes = true;
string playerId = AZ::Uuid::CreateRandom().ToString<string>(includeBrackets,
    includeDashes);
```

Untuk informasi selengkapnya tentang sesi pemain, lihat [Sesi permainan dan pemain di Amazon GameLift Servers konsol](#).

Mesin permainan dan Amazon GameLift Servers

Anda dapat menggunakan yang dikelola Amazon GameLift Servers layanan dengan sebagian besar mesin game utama yang mendukung pustaka C++ atau C#, termasuk O3DE, Unreal Engine, dan Unity. Membangun versi yang Anda butuhkan untuk game Anda; melihat file README dengan setiap versi untuk membangun instruksi dan persyaratan minimum. Untuk informasi lebih lanjut tentang tersedia Amazon GameLift Servers SDKs, platform pengembangan dan sistem operasi yang didukung, lihat [Dapatkan alat Amazon GameLift Servers pengembangan](#) untuk server game.

Selain informasi khusus mesin yang disediakan dalam topik ini, temukan bantuan tambahan untuk mengintegrasikan Amazon GameLift Servers ke server game, klien, dan layanan Anda dalam topik berikut:

- [Tambahkan Amazon GameLift Servers ke server game Anda](#)— Instruksi terperinci tentang pengintegrasian Amazon GameLift Servers ke server game.
- [Tambahkan Amazon GameLift Servers ke klien game Anda](#) – Detail instruksi tentang mengintegrasikan ke dalam klien atau layanan game, termasuk membuat sesi game dan bergabung dengan pemain ke game.

O3DE

Server game

Siapkan server game Anda untuk hosting Amazon GameLift Servers menggunakan SDK [server untuk Amazon GameLift Servers untuk C ++](#). Lihat [Tambahkan Amazon GameLift Servers ke server game Anda](#) untuk mendapatkan bantuan dengan mengintegrasikan fungsionalitas yang diperlukan ke server game Anda.

Klien dan layanan Game

Memungkinkan klien game dan/atau layanan game Anda untuk berinteraksi Amazon GameLift Servers layanan, seperti untuk menemukan sesi permainan yang tersedia atau membuat yang baru, dan menambahkan pemain ke game. fungsionalitas klien inti disediakan di[SDK AWS for C++](#). Untuk mengintegrasikan Amazon GameLift Servers ke dalam proyek game O3DE Anda, lihat [Tambahkan Amazon GameLift Servers ke klien dan server game O3DE](#) dan. [Tambahkan Amazon GameLift Servers ke klien game Anda](#)

Unreal Engine

Server Game

Siapkan server game Anda untuk hosting Amazon GameLift Servers dengan menambahkan SDK [server untuk Amazon GameLift Servers untuk Unreal Engine](#) ke proyek Anda dan menerapkan fungsionalitas server yang diperlukan. Untuk bantuan menyiapkan plugin Unreal Engine dan menambahkan Amazon GameLift Servers kode, lihat[Integrasikan Amazon GameLift Servers ke dalam proyek Unreal Engine](#).

Klien dan layanan Game

Memungkinkan klien game dan/atau layanan game Anda untuk berinteraksi Amazon GameLift Servers layanan, seperti untuk menemukan sesi permainan yang tersedia atau membuat yang baru, dan menambahkan pemain ke game. fungsionalitas klien inti disediakan di[SDK AWS for C++](#). Untuk mengintegrasikan Amazon GameLift Servers ke dalam proyek game Unreal Engine Anda, lihat[Tambahkan Amazon GameLift Servers ke klien game Anda](#).

Unity

Server game

Siapkan server game Anda untuk hosting Amazon GameLift Servers dengan menambahkan SDK [server untuk Amazon GameLift Servers untuk C #](#) ke proyek Anda dan menerapkan fungsionalitas

server yang diperlukan. Untuk bantuan pengaturan dengan Unity dan menambahkan Amazon GameLift Servers kode, lihat [Integrasikan Amazon GameLift Servers ke dalam proyek Unity](#).

Klien dan layanan Game

Memungkinkan klien game dan/atau layanan game Anda untuk berinteraksi Amazon GameLift Servers layanan, seperti untuk menemukan sesi permainan yang tersedia atau membuat yang baru, dan menambahkan pemain ke game. fungsionalitas klien inti disediakan di [AWS SDK untuk .NET](#). Untuk mengintegrasikan Amazon GameLift Servers ke dalam proyek game Unity Anda, lihat [Tambahkan Amazon GameLift Servers ke klien game Anda](#).

Mesin lainnya

Untuk daftar lengkap Amazon GameLift Servers SDKs tersedia untuk server game dan klien, lihat [the section called "Dapatkan alat pengembangan"](#).

Tambahkan Amazon GameLift Servers ke klien dan server game O3DE

Anda dapat menggunakan O3DE, mesin 3D open-source, cross-platform, real time untuk menciptakan pengalaman interaktif berkinerja tinggi, termasuk game dan simulasi. Penyaji dan alat O3DE dibungkus dalam kerangka modular yang dapat Anda modifikasi dan perluas dengan alat pengembangan pilihan Anda.

Kerangka modular menggunakan Permata yang berisi pustaka dengan antarmuka dan aset standar. Pilih Permata Anda sendiri untuk memilih fungsionalitas apa yang akan ditambahkan berdasarkan kebutuhan Anda.

Bagian Amazon GameLift Servers Permata menyediakan fitur-fitur berikut:

Amazon GameLift Servers integrasi

Kerangka kerja untuk memperluas lapisan jaringan O3DE dan membiarkan Permata Multiplayer bekerja dengan Amazon GameLift Servers solusi server khusus. Permata menyediakan integrasi dengan SDK [server untuk Amazon GameLift Servers](#) dan klien AWS SDK (untuk memanggil Amazon GameLift Servers layanan itu sendiri).

Membangun dan manajemen paket

Petunjuk untuk mengemas dan secara opsional mengunggah build server khusus dan aplikasi AWS Cloud Development Kit (AWS CDK) (AWS CDK) untuk menyiapkan dan memperbarui sumber daya.

Amazon GameLift Servers Pengaturan permata

Ikuti prosedur di bagian ini untuk mengatur Amazon GameLift Servers Permata di O3DE.

Prasyarat

- Siapkan AWS akun Anda untuk Amazon GameLift Servers. Untuk informasi selengkapnya, lihat [Mengatur sebuah Akun AWS](#).
- Siapkan AWS kredensial untuk O3DE. Untuk informasi selengkapnya lihat, [Mengonfigurasi AWS Kredensial](#).
- Mengatur AWS CLI dan AWS CDK. Untuk informasi lebih lanjut, [AWS Command Line Interfacedan AWS Cloud Development Kit \(AWS CDK\)](#).

Nyalakan Amazon GameLift Servers Permata dan dependensinya

1. Buka Manajer Proyek.
2. Buka menu di bawah proyek Anda dan pilih Edit Pengaturan Proyek... .
3. Pilih Konfigurasi Permata.
4. Nyalakan Amazon GameLift Servers Permata dan Permata dependen berikut:
 - [AWS Core Gem](#) - Menyediakan kerangka kerja untuk digunakan Layanan AWS di O3DE.
 - [Multiplayer Gem](#) — Menyediakan fungsionalitas multiplayer dengan memperluas kerangka jaringan.

Sertakan Amazon GameLift Servers Perpustakaan statis permata

1. Sertakan `Gem::AWSGameLift.Server.Static` as `BUILD_DEPENDENCIES` untuk target server proyek Anda.

```
ly_add_target(  
  NAME YourProject.Server.Static STATIC  
  ...  
  BUILD DEPENDNCIES  
    PUBLIC  
    ...  
    PRIVATE  
    ...  
    Gem::AWSGameLift.Server.Static
```

```
)
```

2. Setel `AWSGameLiftService` ke `required` untuk komponen sistem server proyek Anda.

```
void
YourProjectServerSystemComponent::GetRequiredServices(AZ::ComponentDescriptor::DependencyArray&
required)
{
    ...
    required.push_back(AZ_CRC_CE("AWSGameLiftServerService"));
    ...
}
```

3. (Opsional) Untuk membuat Amazon GameLift Servers permintaan layanan di C ++, termasuk `Gem::AWSGameLift.Client.Static` dalam `BUILD_DEPENDENCIES` untuk target klien Anda.

```
ly_add_target(
    NAME YourProject.Client.Static STATIC
    ...
    BUILD_DEPENDENCIES
    PUBLIC
    ...
    PRIVATE
    ...
    Gem::AWSCore.Static
    Gem::AWSGameLift.Client.Static
}
```

Integrasikan game dan dedicated server Anda

Kelola sesi permainan dalam game dan server game khusus Anda dengan [Integrasi Manajemen Sesi](#). Untuk mendukung FlexMatch, lihat [FlexMatch Integrasi](#).

Integrasikan Amazon GameLift Servers ke dalam proyek Unreal Engine

Pelajari cara mengintegrasikan Amazon GameLift Servers SDK for Unreal Engine ke dalam project game Anda untuk mengakses set fitur SDK server lengkap.

 Kiat

Untuk penyebaran yang lebih cepat, coba plugin Amazon GameLift Servers mandiri untuk Unreal Engine. Ini menyediakan alur kerja UI terpandu untuk menyebarkan server game Anda dengan cepat dengan pengaturan minimal, sehingga Anda dapat mencoba komponen game Anda dalam aksi. Lihat [Amazon GameLift Servers plugin untuk Unreal Engine](#).

Sumber daya tambahan:

- [Server C ++ \(Tidak nyata\) SDK 5.x untuk -- Tindakan Amazon GameLift Servers](#)
- [Dapatkan alat Amazon GameLift Servers pengembangan](#)

Instal SDK server untuk Unreal

Dapatkan Amazon GameLift Servers SDK sumber terbuka untuk Unreal Engine dari. [GitHub](#) File readme repositori berisi prasyarat dan instruksi instalasi.

Siapkan target build dan aturan modul

Ubah file proyek game Anda untuk menghasilkan komponen build dengan benar untuk digunakan Amazon GameLift Servers.

Untuk menambahkan target build klien dan server:

1. Buka file kode proyek game Anda dan cari file `.../Games/[your application name]Source/[your application name]Target.cs` file. Contoh: `.../Source/GameLiftUnrealAppTarget.cs`. (Jika Anda menggunakan Visual Studio, buka `.sln` file proyek.)
2. Salin file ini untuk membuat dua file target baru di `Source/` direktori.
 - Target klien - Ubah nama file baru menjadi `[your application name]Client.Target.cs`. Edit konten untuk memperbarui nama kelas dan nilai tipe target, seperti yang diilustrasikan dalam kode contoh berikut:

```
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");
    }
}
```

- Target server - Ubah nama file baru menjadi *[your application name]*Server.Target.cs. Edit konten untuk memperbarui nama kelas dan nilai tipe target, seperti yang diilustrasikan dalam kode contoh berikut:

```
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. Regenerasi file proyek Anda. Jika Anda menggunakan Visual Studio, Anda dapat mengklik kanan .uproject file proyek game Anda dan memilih Generate Visual Studio Project Files.

Untuk memperbarui aturan modul proyek game:

Perbarui aturan modul proyek game untuk mengambil ketergantungan pada plugin.

1. Buka file kode proyek game Anda dan cari file *.../Games/[your application name]Source/[your application name].Build.cs* file. Contoh: *.../Source/GameLiftUnrealApp.Build.cs*. (Jika Anda menggunakan Visual Studio, buka .sln file proyek.)

2. Temukan `ModuleRules` kelas dan perbarui seperti yang diilustrasikan dalam kode contoh berikut:

```
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. Setelah membuat file target baru dan memodifikasi aturan modul, bangun kembali proyek game Anda.

Tambahkan fungsionalitas hosting game ke kode server Anda

Setelah instalasi dan penyiapan SDK server, langkah selanjutnya adalah mengintegrasikan fungsionalitas hosting game ke dalam kode server Anda. Server SDK memungkinkan server game Anda untuk berkomunikasi dengan Amazon GameLift Servers layanan, menerima instruksi untuk sesi permainan, melaporkan status dan kesehatan, dan melakukan tindakan lainnya.

Topik ini menyediakan contoh kode yang menambahkan fungsionalitas minimum yang diperlukan untuk meng-host game AndaAmazon GameLift Servers.

Langkah 1: Perbarui file **GameMode** header

1. Buka file kode proyek game Anda dan cari file *Your-application-name*GameMode.h file. Contoh:GameLiftUnrealAppGameMode.h. Jika Anda menggunakan Visual Studio, buka .sln file untuk proyek game Anda.
2. Ubah file header untuk menyertakan kode contoh berikut. Pastikan untuk mengganti "GameLiftUnrealApp" dengan nama aplikasi Anda sendiri.

Contoh kode GameMode untuk.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;
};
```

Langkah 2: Tambahkan panggilan SDK server yang diperlukan ke kode server game Anda

Gunakan kode contoh di bagian ini untuk mengintegrasikan kode server game Anda untuk digunakanAmazon GameLift Servers. Untuk detail tentang apa yang dilakukan kode, lihat [Inisialisasi proses server](#) dan[Server C ++ \(Tidak nyata\) SDK 5.x untuk -- Tindakan Amazon GameLift Servers](#).

Note

Bendera WITH_GAMELIFT preprocessor melayani dua tujuan:

- Membatasi panggilan API Amazon GameLift Servers backend ke build server Unreal saja
- Memastikan kompatibilitas di berbagai target build Unreal

1. Buka file *Your-application-name*GameMode.cpp file sumber terkait. Dalam contoh kita:GameLiftUnrealAppGameMode.cpp.
2. Ubah kode untuk menyelaraskan dengan kode contoh berikut. Pastikan untuk mengganti contoh "GameLiftUnrealApp" dengan nama aplikasi Anda sendiri.

Contoh kode yang disediakan menunjukkan cara menambahkan elemen yang diperlukan untuk integrasi denganAmazon GameLift Servers. Ini termasuk:

- Inisialisasi klien Amazon GameLift Servers API.
- Menerapkan fungsi callback untuk menanggapi permintaan dari Amazon GameLift Servers layanan, termasukOnStartGameSession,OnProcessTerminate, danonHealthCheck.
- Hubungi ProcessReady() untuk memberi tahu Amazon GameLift Servers layanan saat siap menyelenggarakan sesi permainan.

Contoh kode GameMode untuk.cpp

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

#include "GameLiftUnrealAppGameMode.h"

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

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

#include "GenericPlatform/GenericPlatformOutputDevices.h"

DEFINE_LOG_CATEGORY(GameServerLog);

AGameLiftUnrealAppGameMode::AGameLiftUnrealAppGameMode() :
    ProcessParameters(nullptr)
```

```

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

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

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

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

#ifdef WITH_GAMELIFT
    InitGameLift();
#endif
}

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

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

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

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

    if (bIsAnywhereActive)
    {

```



```

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

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

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

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

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

    FString glAnywhereAuthToken = "";

```

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

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

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

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

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

        UE_LOG(GameServerLog, SetColor, TEXT("%s"), COLOR_YELLOW);
        UE_LOG(GameServerLog, Log, TEXT(">>>> WebSocket URL: %s"),
*ServerParametersForAnywhere.m_webSocketUrl);
        UE_LOG(GameServerLog, Log, TEXT(">>>> Fleet ID: %s"),
*ServerParametersForAnywhere.m_fleetId);
```

```

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

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

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

    ProcessParameters = MakeShared<FProcessParameters>();

    //When a game session is created, Amazon GameLift Servers sends an activation
request to the game server and passes along the game session object containing game
properties and other settings.

```

```

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

//OnProcessTerminate callback. Amazon GameLift Servers will invoke this callback
before shutting down an instance hosting this game server.
//It gives this game server a chance to save its state, communicate with services,
etc., before being shut down.
//In this case, we simply tell Amazon GameLift Servers we are indeed going to
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  
}
```

Langkah 3: Membangun kembali proyek game

- Bangun proyek game untuk kedua jenis target berikut: Editor Pengembangan dan Server Pengembangan.

Note

Anda tidak perlu membangun kembali solusinya. Sebagai gantinya, buat project di bawah /Games/ folder untuk aplikasi Anda. Jika tidak, Visual Studio akan membangun kembali seluruh UE5 proyek, yang dapat memakan waktu hingga satu jam.

Package server game Anda untuk hosting

Dengan kode server game Anda yang sekarang terintegrasi dengan fungsionalitas SDK server minimum yang diperlukan, Anda siap untuk mengemas build server game Anda menggunakan Unreal Editor.

Untuk mengemas build server game

1. Buka proyek game di Unreal Editor.
2. Ikuti langkah-langkah Unreal Editor untuk mengemas server game Anda:
 - Pilih platform target Anda (Windows atau Linux).
 - Pilih target build server Anda (*[your application name]*Server).

Proses pengemasan menghasilkan server game Anda yang dapat dieksekusi: *[your application name]*Server.exe

3. Siapkan build server game Anda untuk penyebaran ke sumber daya hosting. Build harus menyertakan file-file berikut:
 - Server game Anda dapat dieksekusi

- Untuk build Windows, sertakan file-file ini (Anda dapat menemukannya di versi Unreal Engine yang dibuat sumber):
 - VC_redist.x64.exe (UnrealEngine\Engine\Source\Programs\PrereqInstaller\Resources\VCRedist\)
 - UEPrereqSetup_x64.exe or UE5PrereqSetup_x64.exe (UnrealEngine\Engine\Extras\Redist\en-us\)
- Semua dependensi lain yang diperlukan untuk server game Anda.
- Pustaka OpenSSL, jika diperlukan. Periksa versi SDK server yang terintegrasi dengan build server game Anda. Dengan SDK server untuk Unreal, versi 5.3.x, Anda tidak perlu lagi menyertakan pustaka OpenSSL secara manual ke dalam build server game Anda. Untuk semua versi sebelumnya, Anda harus mencari dan menambahkan file perpustakaan. [Lihat versi SDK server terbaru.](#)

Pustaka OpenSSL diperlukan untuk server game yang dibuat dengan server SDK untuk Unreal, versi 5.2.x dan versi sebelumnya

Anda harus menyertakan versi pustaka OpenSSL yang sama yang digunakan saat mengemas server game di Unreal. Perpustakaan ini terletak di sumber mesin game Anda. Lokasi bervariasi tergantung pada lingkungan pengembangan Anda:

Di Windows:

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

Di 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

Salin pustaka OpenSSL ke paket build game Anda di direktori yang sama dengan file yang dapat dieksekusi server game.

Langkah selanjutnya

Sekarang setelah Anda menyiapkan build server game dengan fungsionalitas minimum yang diperlukan untuk hosting Amazon GameLift Servers, pertimbangkan langkah-langkah selanjutnya yang potensial ini:

- Menyebarkan server game terintegrasi Anda untuk dan pengujian dan pengembangan. Dengan armada Anywhere, Anda dapat mengatur mesin lokal Anda sebagai sumber daya hosting dan menggunakannya untuk menguji server game dan koneksi klien game Anda. Untuk hosting berbasis cloud, terapkan server game Anda ke armada kontainer terkelola EC2 atau terkelola. Lihat topik ini untuk panduan:
 - [Siapkan untuk pengembangan berulang dengan Amazon GameLift Servers Dimanapun](#)
 - [Amazon GameLift Servers Armada di mana saja](#)
 - [Amazon GameLift Servers EC2 armada terkelola](#)
 - [Amazon GameLift Servers armada kontainer terkelola](#)
- Sesuaikan integrasi server game Anda dengan menambahkan fitur opsional. Misalnya, Anda mungkin ingin menambahkan sesi pemain dengan pemain unik IDs, menyiapkan pengisian ulang perijinan, atau mengelola akses server game ke AWS sumber daya Anda yang lain (seperti database atau layanan penyimpanan konten). Lihat topik ini untuk panduan:
 - [Tambahkan Amazon GameLift Servers ke server game Anda](#)
 - [Server C ++ \(Tidak nyata\) SDK 5.x untuk -- Tindakan Amazon GameLift Servers](#)
- Sesuaikan komponen klien game Anda untuk meminta sesi game, menerima informasi koneksi, dan terhubung langsung ke server game untuk memainkan game. Lihat topik ini untuk panduan:
 - [Integrasikan klien game Anda dengan Amazon GameLift Servers](#)

Integrasikan Amazon GameLift Servers ke dalam proyek Unity

Pelajari cara mengintegrasikan Amazon GameLift Servers SDK for Unity ke dalam project game Anda untuk mengakses set fitur SDK server lengkap.

Kiat

Untuk penerapan yang lebih cepat, coba plugin Amazon GameLift Servers mandiri untuk Unity. Ini menyediakan alur kerja UI terpandu untuk menyebarkan server game Anda dengan

cepat dengan pengaturan minimal, sehingga Anda dapat mencoba komponen game Anda dalam aksi. Lihat [Amazon GameLift Serversplugin untuk Unity \(server SDK 5.x\)](#).

Sumber daya tambahan:

- [C# server SDK 5.x untuk Amazon GameLift Servers -- Tindakan](#)
- [the section called “Dapatkan alat pengembangan”](#)

Instal SDK for Unity server

Dapatkan sumber terbuka Amazon GameLift Servers untuk Unity dari [GitHub](#). File readme repositori berisi prasyarat dan instruksi instalasi.

Siapkan armada Amazon GameLift Servers Anywhere untuk pengujian

Anda dapat mengatur workstation pengembangan Anda sebagai armada hosting Amazon GameLift Servers Anywhere untuk menguji integrasi Anda Amazon GameLift Servers secara berulang. Dengan pengaturan ini, Anda dapat memulai proses server game di workstation Anda, mengirim permintaan bergabung pemain atau perjodohan Amazon GameLift Servers untuk memulai sesi permainan, dan menghubungkan klien ke sesi permainan baru. Dengan workstation Anda sendiri yang diatur sebagai server hosting, Anda dapat memantau semua aspek integrasi game Anda. Amazon GameLift Servers

Untuk petunjuk tentang pengaturan workstation Anda, lihat [Siapkan pengujian lokal dengan Amazon GameLift Servers Anywhere](#) untuk menyelesaikan langkah-langkah berikut:

1. Buat lokasi khusus untuk workstation Anda.
2. Buat armada Amazon GameLift Servers Anywhere dengan lokasi kustom baru Anda. Jika berhasil, permintaan ini mengembalikan ID armada. Catat nilai ini, karena Anda akan membutuhkannya nanti.
3. Daftarkan workstation Anda sebagai komputasi di armada Anywhere yang baru. Berikan nama komputasi yang unik dan tentukan alamat IP untuk workstation Anda. Jika berhasil, permintaan ini mengembalikan titik akhir SDK layanan, dalam bentuk URL. WebSocket Catat nilai ini, karena Anda akan membutuhkannya nanti.
4. Hasilkan token otentikasi untuk komputasi workstation Anda. Otentikasi berumur pendek ini mencakup token dan tanggal kedaluwarsa. Server game Anda menggunakannya untuk mengautentikasi komunikasi dengan Amazon GameLift Servers layanan. Simpan otentikasi pada

komputasi workstation Anda sehingga proses server game Anda yang sedang berjalan dapat mengaksesnya.

Tambahkan kode Amazon GameLift Servers server ke proyek Unity Anda

Server game Anda berkomunikasi dengan Amazon GameLift Servers layanan untuk menerima instruksi dan melaporkan status yang sedang berlangsung. Untuk mencapai hal ini, Anda menambahkan kode server game yang menggunakan SDK Amazon GameLift Servers server.

Contoh kode yang disediakan menggambarkan elemen integrasi dasar yang diperlukan. Ini menggunakan a `MonoBehavior` untuk mengilustrasikan inisialisasi server game sederhana dengan. Amazon GameLift Servers Contohnya mengasumsikan bahwa server game berjalan pada armada Amazon GameLift Servers Anywhere untuk pengujian. Ini termasuk kode untuk:

- Inisialisasi klien Amazon GameLift Servers API. Sampel menggunakan versi `InitSDK()` dengan parameter server untuk armada dan komputasi Anywhere Anda. Gunakan WebSocket URL, ID armada, nama komputasi (ID host), dan token otentikasi, seperti yang didefinisikan dalam topik sebelumnya. [Siapkan armada Amazon GameLift Servers Anywhere untuk pengujian](#)
- Menerapkan fungsi callback untuk menanggapi permintaan dari Amazon GameLift Servers layanan, termasuk `OnStartGameSession`, `OnProcessTerminate`, dan `onHealthCheck`.
- Panggil `ProcessReady()` dengan port yang ditunjuk untuk memberi tahu Amazon GameLift Servers layanan ketika proses siap untuk menyelenggarakan sesi permainan.

Kode sampel yang diberikan menetapkan komunikasi dengan Amazon GameLift Servers layanan. Ini juga mengimplementasikan satu set fungsi callback yang merespons permintaan dari layanan. Amazon GameLift Servers Untuk informasi selengkapnya tentang setiap fungsi dan apa yang dilakukan kode, lihat [Menginisialisasi proses server](#). Untuk informasi selengkapnya tentang tindakan SDK dan tipe data yang digunakan dalam kode ini, baca [C# server SDK 5.x untuk Amazon GameLift Servers -- Tindakan](#).

Kode contoh menunjukkan cara menambahkan fungsionalitas yang diperlukan, seperti yang dijelaskan dalam [Tambahkan Amazon GameLift Servers ke server game Anda](#). Untuk informasi selengkapnya tentang tindakan SDK server, lihat. [C# server SDK 5.x untuk Amazon GameLift Servers -- Tindakan](#)

Kode sampel integrasi

```
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);
}
}
```

Langkah selanjutnya

Sekarang setelah Anda menyiapkan build server game dengan fungsionalitas minimum yang diperlukan untuk hosting Amazon GameLift Servers, pertimbangkan langkah-langkah selanjutnya yang potensial ini:

- Menyebarkan server game terintegrasi Anda untuk dan pengujian dan pengembangan. Dengan armada Anywhere, Anda dapat mengatur mesin lokal Anda sebagai sumber daya hosting dan menggunakannya untuk menguji server game dan koneksi klien game Anda. Untuk hosting berbasis cloud, terapkan server game Anda ke armada kontainer terkelola EC2 atau terkelola. Lihat topik ini untuk panduan:
 - [Siapkan untuk pengembangan berulang dengan Amazon GameLift Servers Dimanapun](#)
 - [Amazon GameLift Servers Armada di mana saja](#)
 - [Amazon GameLift Servers EC2 armada terkelola](#)
 - [Amazon GameLift Servers armada kontainer terkelola](#)
- Sesuaikan integrasi server game Anda dengan menambahkan fitur opsional. Misalnya, Anda mungkin ingin menambahkan sesi pemain dengan pemain unik IDs, menyiapkan pengisian ulang perijodohan, atau mengelola akses server game ke AWS sumber daya Anda yang lain (seperti database atau layanan penyimpanan konten). Lihat topik ini untuk panduan:
 - [Tambahkan Amazon GameLift Servers ke server game Anda](#)
 - [Server C++ \(Tidak nyata\) SDK 5.x untuk -- Tindakan Amazon GameLift Servers](#)
- Sesuaikan komponen klien game Anda untuk meminta sesi game, menerima informasi koneksi, dan terhubung langsung ke server game untuk memainkan game. Lihat topik ini untuk panduan:
 - [Integrasikan klien game Anda dengan Amazon GameLift Servers](#)

Rancang layanan klien game Anda

Kami menyarankan Anda menerapkan layanan klien game yang mengotentikasi pemain Anda dan berkomunikasi dengan Amazon GameLift Servers API. Dengan menerapkan layanan klien game khusus, Anda dapat:

- Sesuaikan otentikasi untuk pemain Anda.
- Kontrol bagaimana Amazon GameLift Servers pertandingan dan memulai sesi permainan.
- Gunakan database pemain Anda untuk atribut pemain seperti peringkat keterampilan untuk perjodohan alih-alih mempercayai klien.

Menggunakan layanan klien game juga mengurangi risiko keamanan yang diperkenalkan oleh klien game yang berinteraksi langsung dengan Anda Amazon GameLift Servers API.

Mengautentikasi pemain Anda

Anda dapat menggunakan Amazon Cognito dan sesi pemain IDs untuk mengautentikasi klien game Anda. Untuk mengelola siklus hidup dan properti identitas pemain Anda, gunakan kumpulan pengguna Amazon Cognito.

Jika Anda mau, buat solusi identitas khusus dan host AWS. Anda juga dapat menggunakan otorisasi Lambda untuk logika otorisasi kustom dengan API Gateway.

Sumber daya tambahan:

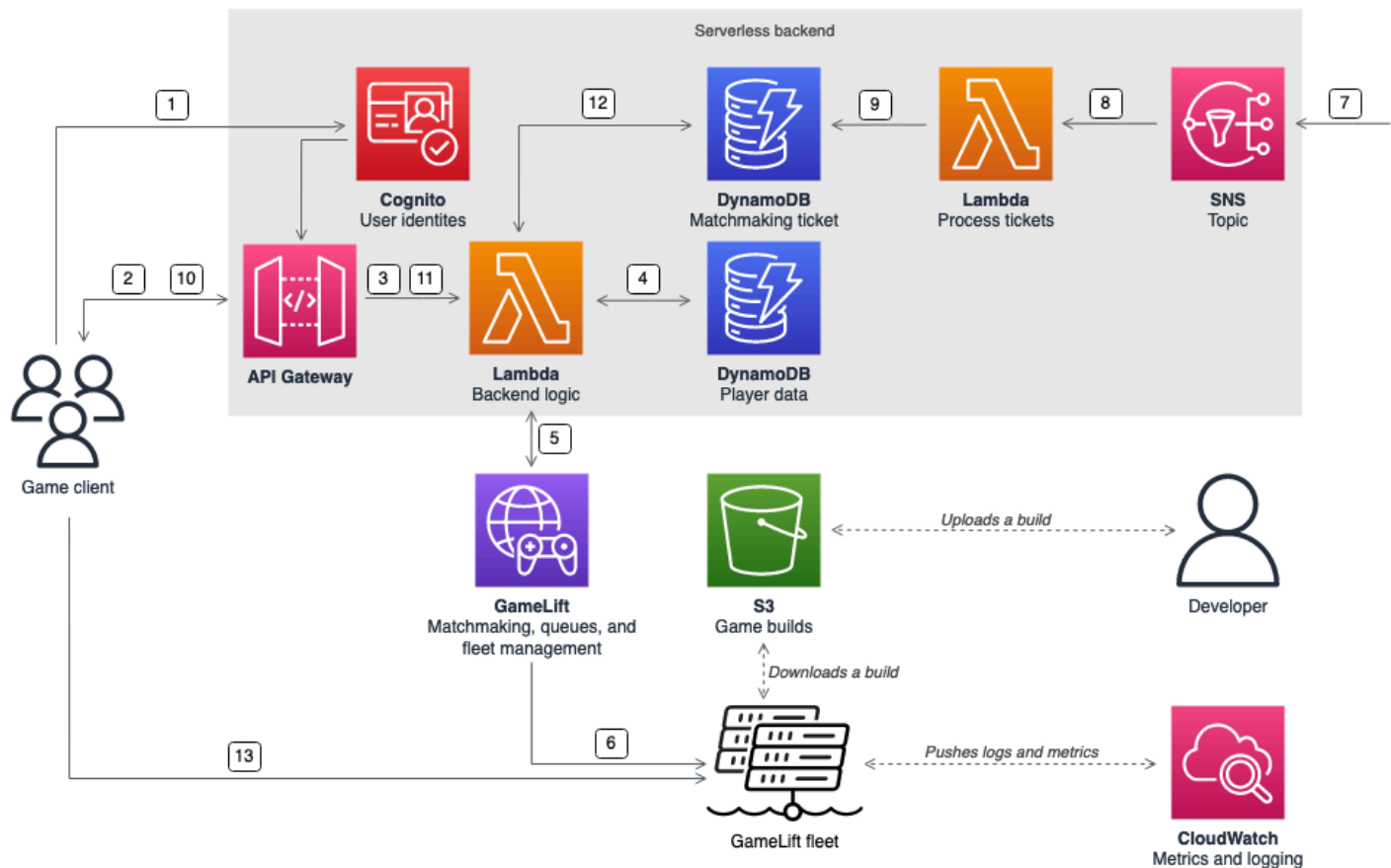
- [Menggunakan kumpulan identitas \(identitas federasi\) \(Panduan Pengembang Amazon Cognito\)](#)
- [Memulai kumpulan pengguna](#) (Panduan Pengembang Amazon Cognito)
- [Cara Mengatur Otentikasi Pemain dengan Amazon](#) Cognito AWS (untuk Blog Game)

Server sesi game mandiri dengan backend tanpa server

Menggunakan arsitektur layanan klien tanpa server, backend dapat melihat status tiket perjodohan dari database yang sangat skalabel, bukan dengan langsung mengakses Amazon GameLift Servers API.

Diagram berikut menunjukkan backend tanpa server yang dibangun dengan yang cocok dengan pemain ke dalam game Layanan AWS yang berjalan Amazon GameLift Servers armada. Daftar

berikut memberikan deskripsi untuk setiap callout bernomor dalam diagram. Untuk mencoba contoh ini, lihat [Hosting Game Berbasis Sesi Multiplayer](#) di AWS GitHub



1. Klien game meminta identitas pengguna Amazon Cognito dari kumpulan identitas Amazon Cognito.
2. Klien game menerima kredensial akses sementara dan meminta sesi game melalui API Amazon API Gateway.
3. API Gateway memanggil AWS Lambda fungsi.
4. Fungsi Lambda meminta data pemutar dari tabel NoSQL Amazon DynamoDB. Fungsi ini menyediakan identitas Amazon Cognito dalam data konteks permintaan.
5. Fungsi Lambda meminta kecocokan melalui Amazon GameLift Servers FlexMatch perijodohan.
6. FlexMatch mencocokkan sekelompok pemain dengan latensi yang sesuai, dan kemudian meminta penempatan sesi permainan melalui Amazon GameLift Servers antrian. Antrian memiliki armada dengan satu atau lebih Wilayah AWS lokasi di dalamnya.

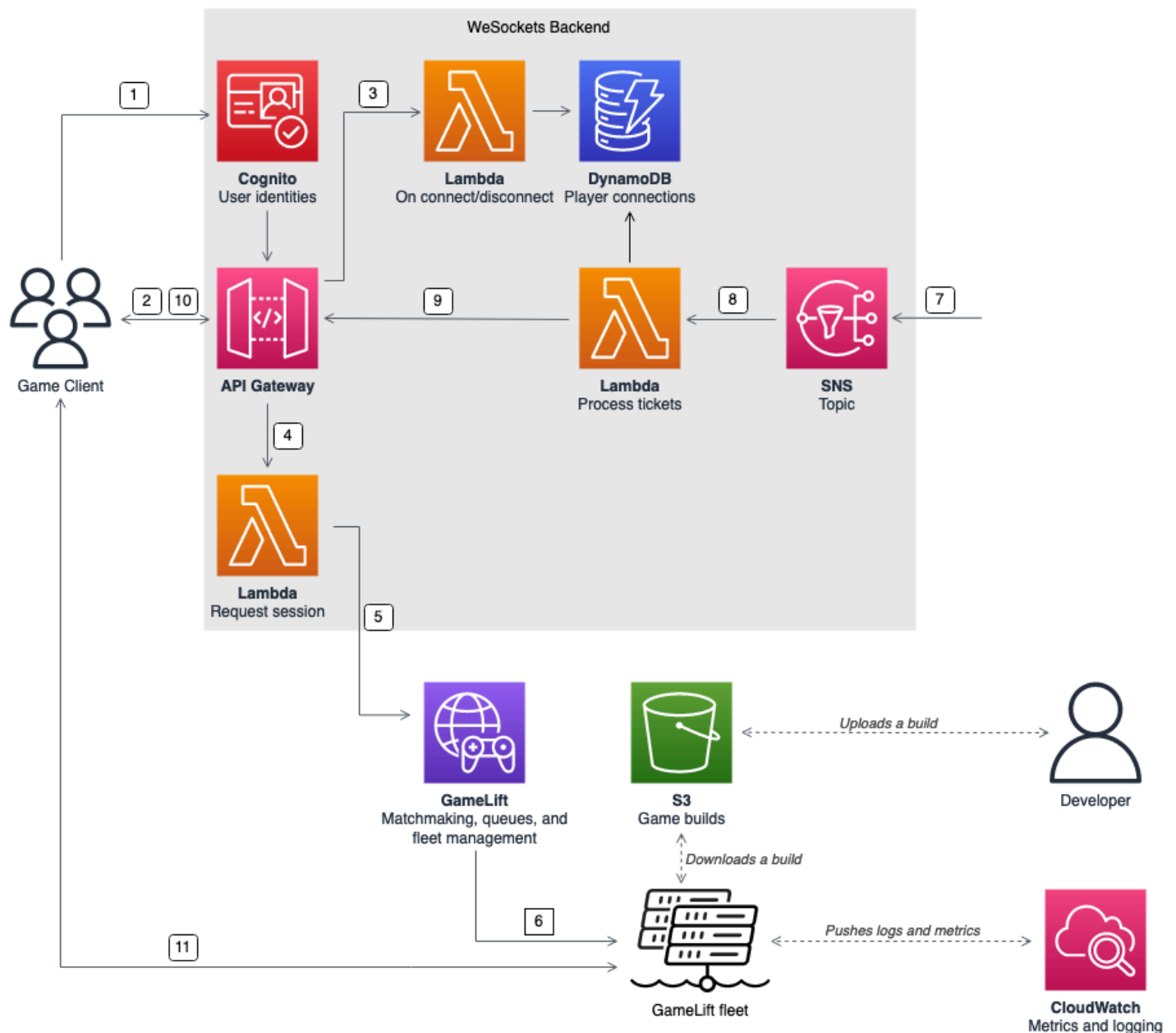
7. Setelah Amazon GameLift Servers menempatkan sesi di salah satu lokasi armada, Amazon GameLift Servers mengirimkan pemberitahuan acara ke topik Amazon Simple Notification Service (Amazon SNS).
8. Fungsi Lambda menerima acara Amazon SNS dan memprosesnya.
9. Jika tiket perjodohan adalah sebuah MatchmakingSucceeded acara, maka fungsi Lambda menulis hasilnya, bersama dengan port dan alamat IP server game, ke tabel DynamoDB.
- 10Klien game membuat permintaan yang ditandatangani ke API Gateway untuk melihat status tiket perjodohan pada interval tertentu.
- 11API Gateway menggunakan fungsi Lambda yang memeriksa status tiket perjodohan.
- 12Fungsi Lambda memeriksa tabel DynamoDB untuk melihat apakah tiket berhasil. Jika telah berhasil, fungsi mengirimkan port server game dan alamat IP, bersama dengan ID sesi pemain, kembali ke klien. Jika tiket belum berhasil, fungsi mengirimkan respons yang memverifikasi bahwa kecocokan belum siap.
- 13Klien game terhubung ke server game menggunakan TCP atau UDP dengan menggunakan port dan alamat IP yang disediakan layanan backend. Klien game kemudian mengirimkan ID sesi pemain ke server game, yang kemudian memvalidasi ID menggunakan SDK server untuk Amazon GameLift Servers.

Server sesi game mandiri dengan backend WebSocket berbasis

Menggunakan arsitektur WebSocket berbasis Amazon API Gateway, Anda dapat membuat permintaan perjodohan dengan WebSockets dan mengirim pemberitahuan push untuk penyelesaian perjodohan menggunakan pesan yang dimulai server. Arsitektur ini meningkatkan kinerja dengan memiliki komunikasi dua arah antara klien dan server.

Untuk informasi selengkapnya tentang menggunakan API Gateway WebSock APIs, lihat [Bekerja dengan WebSocket APIs](#).

Diagram berikut menunjukkan arsitektur backend WebSocket berbasis yang menggunakan API Gateway dan lainnya Layanan AWS untuk mencocokkan pemain ke dalam game yang sedang berjalan Amazon GameLift Servers armada. Daftar berikut memberikan deskripsi untuk setiap callout bernomor dalam diagram.



1. Klien game meminta identitas pengguna Amazon Cognito dari kumpulan identitas Amazon Cognito.
2. Klien game menandatangani WebSocket koneksi ke API Gateway API dengan kredensial Amazon Cognito.
3. API Gateway memanggil AWS Lambda fungsi pada koneksi. Fungsi menyimpan informasi koneksi dalam tabel Amazon DynamoDB.
4. Klien game mengirim pesan ke fungsi Lambda, melalui API Gateway API melalui WebSocket koneksi, untuk meminta sesi.

5. Fungsi Lambda menerima pesan dan kemudian meminta kecocokan melalui Amazon GameLift Servers FlexMatch perijodohan.
6. Setelah FlexMatch cocok dengan sekelompok pemain, FlexMatch meminta penempatan sesi permainan melalui Amazon GameLift Servers antrian.
7. Setelah Amazon GameLift Servers menempatkan sesi di salah satu lokasi armada, Amazon GameLift Servers mengirimkan pemberitahuan acara ke topik Amazon Simple Notification Service (Amazon SNS).
8. Fungsi Lambda menerima acara Amazon SNS dan memprosesnya.
9. Jika tiket perijodohan adalah MatchmakingSucceeded acara, maka fungsi Lambda meminta koneksi pemain yang benar dari DynamoDB. Fungsi tersebut kemudian mengirimkan pesan ke klien game melalui API Gateway API melalui WebSocket koneksi. Dalam arsitektur ini, klien game tidak secara aktif melakukan polling status perijodohan.
- 10 Klien game menerima port dan alamat IP server game, bersama dengan ID sesi pemain, melalui WebSocket koneksi.
- 11 Klien game terhubung ke server game menggunakan TCP atau UDP menggunakan port dan alamat IP yang disediakan layanan backend. Klien game juga mengirimkan ID sesi pemain ke server game, yang kemudian memvalidasi ID menggunakan SDK server untuk Amazon GameLift Servers.

Siapkan untuk pengembangan berulang dengan Amazon GameLift Servers Dimanapun

Amazon GameLift Servers menyediakan alat dan solusi untuk membantu Anda menyiapkan lingkungan pengujian yang dihosting untuk digunakan selama pengembangan game. Dengan alat ini, Anda dapat menciptakan lingkungan yang mencerminkan pengalaman pemain dunia nyata dari hosting terkelola Amazon GameLift Servers dan mendukung proses pengembangan yang cepat dan berulang.

Dengan lingkungan pengujian terpisah, Anda menghapus overhead Amazon GameLift Servers armada yang dikelola selama pengujian. Anda tidak perlu lagi mengunggah setiap iterasi build server game baru, membuat armada baru untuk itu, dan kemudian menunggu 15+ menit untuk mengaktifkannya. Sebagai gantinya, Anda dapat membuat build baru, memperbarui armada pengujian dengan cepat dengan build baru, memulainya, dan memulai pengujian.

Menggunakan sebuah Amazon GameLift Servers Di mana saja armada, Anda dapat mengatur lingkungan pengujian menggunakan perangkat lokal, seperti workstation pengembangan Anda. Anda juga dapat menyiapkan lingkungan pengujian menggunakan sumber daya hosting berbasis cloud.

Siapkan lingkungan pengujian Anywhere untuk mengembangkan dan menguji berbagai skenario, termasuk ini:

- Uji integrasi server game Anda dengan Amazon GameLift Servers server SDK. Anda dapat menguji bahkan tanpa klien game yang berfungsi dengan menggunakan panggilan AWS CLI untuk memulai sesi permainan baru dan melacak acara sesi game.
- Uji interaksi antara klien game Anda, layanan backend, dan Amazon GameLift Servers layanan saat Anda mengembangkan komponen untuk permainan Anda. Sempurnakan pengalaman pemain untuk bergabung dengan game.
- Bereksperimenlah dengan FlexMatch desain mak comblang. Cobalah variasi set aturan dan implementasi fitur perjodohan lainnya. Siapkan dan uji isi ulang perjodohan.
- Cobalah yang lain Amazon GameLift Servers fitur hosting, seperti pengaturan konfigurasi runtime (dengan Amazon GameLift Servers Agen) untuk manajemen siklus hidup server game.
- Bangun, uji, dan ulangi dengan cepat untuk memvalidasi semua aspek pengalaman pemain game Anda, termasuk interaksi multipemain, di lingkungan langsung yang dihosting.

Nanti, saat Anda mempersiapkan game untuk diluncurkan, Anda pasti ingin menambahkan Amazon GameLift Servers armada terkelola untuk menyempurnakan konfigurasi hosting Anda dan menguji skenario tambahan, termasuk yang berikut:

- Bereksperimenlah dengan dan uji desain antrian sesi permainan, termasuk penggunaan armada multi-lokasi, armada Spot dan On-Demand, dan beberapa jenis instans.
- Cobalah opsi penempatan sesi game dengan armada terkelola, termasuk penggunaan kebijakan latensi opsional dan pengaturan prioritas armada.
- Konfigurasi penskalaan kapasitas untuk memenuhi permintaan pemain, menggunakan opsi penskalaan otomatis atau manual.
- Mengatur AWS CloudFormation dengan Amazon GameLift Servers armada terkelola untuk mengelola sumber daya hosting Anda dalam jangka panjang.

Alat Pembaruan Bangun Cepat (hanya untuk pengembangan)

Dengan EC2 armada terkelola, untuk menerapkan pembaruan build server game, Anda perlu mengunggah setiap build baru Amazon GameLift Servers dan membuat armada baru untuk itu.

Fast Build Update Tool memungkinkan Anda dapat melewati langkah-langkah ini selama pengembangan, menghemat waktu dan memungkinkan iterasi pengembangan yang lebih cepat. Dengan alat ini, Anda dapat dengan cepat memperbarui file build game Anda di semua komputasi dalam armada yang ada. Alat ini memiliki beberapa opsi; Anda dapat mengganti seluruh game build atau mengubah 6 file tertentu, dan Anda dapat mengelola cara memulai ulang proses server game setelah pembaruan. Anda juga dapat menggunakannya untuk memperbarui komputasi individu dalam armada.

Untuk mendapatkan Fast Build Update Tool dan mempelajari lebih lanjut tentang cara menggunakannya, kunjungi Amazon GameLift Servers Repo Toolkit untuk [Alat Pembaruan Bangun Cepat](#) di Github.

Topik

- [Membangun lingkungan pengujian berbasis cloud](#)
- [Siapkan pengujian lokal dengan Amazon GameLift Servers Anywhere](#)
- [Bekerja dengan Amazon GameLift Servers Agen](#)

Membangun lingkungan pengujian berbasis cloud

Note

Topik ini mencakup pengujian berulang untuk game yang terintegrasi dengan SDK server untuk Amazon GameLift Servers versi 5.x. Jika game Anda menggunakan server SDK versi 4.x atau yang lebih lama, lihat. [Uji integrasi Anda menggunakan Amazon GameLift Servers Lokal:](#)

Gunakan sebuah Amazon GameLift Servers Armada di mana saja untuk membangun dan menguji komponen game Anda secara berulang di lingkungan yang dihosting berbasis cloud. Buat armada Anywhere dengan sumber daya hosting dan koneksi ke Amazon GameLift Servers layanan, jalankan server game Anda di dalamnya, dan uji fungsionalitas game sesuai kebutuhan.

Menyebarkan armada Anywhere dengan Amazon GameLift Servers Agen

Jika build server game Anda terintegrasi dengan Amazon GameLift Servers SDK 5.x atau yang lebih baru, Anda dapat menerapkannya ke armada Anywhere berbasis cloud dengan Amazon GameLift Servers Agen. Agen adalah proses latar belakang yang mengelola siklus hidup server game dan tugas lain pada setiap komputasi dalam armada. Tugas-tugas ini termasuk mendaftarkan komputasi dengan armada Anywhere, memperoleh token otentikasi, dan memulai/menghentikan proses server game berdasarkan serangkaian instruksi. Agen dikendalikan oleh konfigurasi runtime armada, yang dapat Anda perbarui kapan saja selama masa pakai armada. (Agen secara otomatis dikerahkan ke EC2 armada yang dikelola.) Untuk informasi selengkapnya dan mengunduh Agen, lihat [Amazon GameLift Servers GitHubrepository](#).

Siapkan pengujian berulang dengan Amazon EC2

Gunakan alur kerja terpandu dalam hal ini [Amazon GameLift Servers solusi toolkit](#) untuk menyiapkan lingkungan hosting berbasis cloud yang mencerminkan pengalaman hosting terkelola Amazon GameLift Servers.

GitHub Repository menyediakan satu set skrip yang mengotomatiskan sebagian besar proses untuk menyiapkan lingkungan pengujian Amazon GameLift Servers Di mana saja dan Amazon GameLift Servers Agen. Ini juga memberikan panduan untuk memperbarui lingkungan setiap kali Anda memiliki server game baru yang dibangun untuk diuji. Anda dapat menjalankan satu skrip yang menyebarkan lingkungan pengujian dengan contoh pembuatan server game, atau Anda dapat menelusuri setiap langkah untuk mengaturnya dengan build server game Anda sendiri.

Dalam alur kerja ini, Anda akan bekerja sepenuhnya di AWS Management Console, menggunakan AWS CloudShell untuk menjalankan skrip dan menyelesaikan tugas baris perintah.

Note

Untuk tugas-tugas dalam tutorial ini, Anda memerlukan pengguna AWS akun dengan izin untuk layanan berikut: Amazon GameLift Servers, AWS CloudShell, Amazon S3,, AWS Systems Manager Amazon EC2, dan. AWS Identity and Access Management Pengguna dengan akses tingkat admin ke AWS akun sudah memiliki izin yang diperlukan.

Alur kerja mencakup tugas-tugas berikut:

- Package build server game untuk Amazon GameLift Servers. Alur kerja menyediakan skrip untuk membangun contoh server game C++, yang telah terintegrasi dengan SDK server untuk Amazon

GameLift Servers versi 5.x dan siap untuk hosting. Atau, Anda dapat bekerja dengan proyek game Anda sendiri jika Anda telah menyelesaikan integrasi.

- Siapkan bucket Amazon Simple Storage Service untuk menyimpan build dan dependensi server game. Saat Anda menghasilkan versi baru dari build game Anda, Anda dapat menyimpannya di S3 dan menggunakan skrip untuk memperbarui armada Anywhere untuk pengujian game.
- Dapatkan dan bangun Amazon GameLift Servers Agen. Agen mengelola proses server game pada sumber daya hosting berdasarkan konfigurasi Anda. Ia menggunakan logika yang sama dan berperilaku identik Amazon GameLift Servers EC2 hosting terkelola.
- Siapkan armada Anywhere untuk sumber daya hosting Anda. Dengan armada Anywhere Anda dapat menggunakan Amazon GameLift Servers layanan untuk sumber daya hosting yang tidak dikelola oleh Amazon GameLift Servers. Pada langkah ini, Anda juga akan mengonfigurasi konfigurasi runtime, yang menginstruksikan Amazon GameLift Servers Agen kapan dan bagaimana memulai proses server game.
- Siapkan EC2 instans Amazon. Ini adalah lingkungan pengujian Anda untuk pengujian berulang. Jauh lebih cepat untuk menggunakan EC2 instance standar daripada yang dikelola sepenuhnya Amazon GameLift Servers instance (yang dioptimalkan untuk penggunaan tingkat produksi). Dengan EC2 contoh standar, Anda dapat dengan cepat dan terus memperbarui server game sesuai kebutuhan.
- Terapkan build server game Anda dan Amazon GameLift Servers Agen untuk EC2 contoh Amazon. Alur kerja menyediakan skrip yang mendapatkan versi terbaru dari build game Anda dan semua dependensi dan menginstalnya di instance Anda. EC2 Dalam alur kerja ini, dependensi mencakup Amazon GameLift Servers Agen dan CloudWatch Agen
- Mulai Amazon GameLift Servers Agen. Setelah diinstal, Agen secara otomatis memulai dan mulai menjalankan instruksi. Ini termasuk:
 - Daftarkan EC2 instance sebagai komputasi di Amazon GameLift Servers Armada di mana saja.
 - Membangun WebSocket koneksi dengan Amazon GameLift Servers servis dan dapatkan konfigurasi runtime terbaru.
 - Memulai proses server game berdasarkan instruksi dalam konfigurasi runtime. Dalam alur kerja ini, Agen diinstruksikan untuk memulai satu proses server game yang dapat dieksekusi.
- Uji skenario permainan Anda. Dengan pengaturan lingkungan pengujian dan build server game terbaru Anda diinstal, Anda dapat memulai pengujian. Alur kerja berjalan melalui beberapa langkah untuk pengujian termasuk memulai sesi permainan. Akses log server CloudWatch game untuk melacak kemajuan saat sesi permainan dimulai dan bersiap untuk menerima pemain.

Saat Anda mengembangkan komponen game Anda, termasuk klien game dan layanan backend sisi klien, Anda dapat memasukkannya ke dalam skenario pengujian Anda. Gunakan klien game untuk meminta sesi permainan, mengambil info koneksi dari Amazon GameLift Servers layanan, dan kemudian terhubung langsung ke sesi permainan.

- Terapkan build server game baru dan ulangi tes. Saat Anda mengembangkan game Anda, Anda dapat menghasilkan build server game baru, lalu dengan cepat menerapkannya ke lingkungan EC2 pengujian untuk pengujian. Unggah ke bucket Amazon S3 lalu gunakan skrip alur kerja untuk memperbarui lingkungan pengujian.

Transisi permainan Anda ke Amazon GameLift Servers armada terkelola

Setelah Anda menyelesaikan pengujian pengembangan dan Anda siap untuk mempersiapkan peluncuran, ini adalah saat yang tepat untuk beralih ke Amazon GameLift Servers armada yang dikelola. Gunakan armada terkelola untuk menyempurnakan dan menguji sumber daya hosting game Anda. Terapkan solusi penempatan sesi permainan Anda (antrian dan mak comblang), pilih perangkat keras hosting optimal (termasuk armada Spot) dan lokasi, dan pilih strategi untuk kapasitas penskalaan. Anda mungkin juga ingin mulai menggunakan AWS CloudFormation untuk mengelola siklus hidup semua sumber daya hosting game Anda dengan lebih efisien, termasuk armada, antrian, dan mak comblang.

Dibutuhkan upaya minimal untuk transisi dari armada uji Anywhere berbasis cloud ke Amazon GameLift Servers armada yang dikelola. Anda tidak perlu mengubah kode permainan apa pun, dan Anda dapat menggunakan kembali antrian dan mak comblang yang sama. Lakukan tugas-tugas berikut:

- Buat sebuah Amazon GameLift Servers membangun sumber daya. Dengan armada pengujian Anywhere, Anda harus menerapkan build dan dependensi server game secara manual ke setiap komputasi armada. Dengan armada terkelola, unggah paket pembuatan game Anda ke Amazon GameLift Servers, yang secara otomatis menyebarkannya ke semua komputasi armada. Lihat [Menerapkan build server khusus untuk Amazon GameLift Servers mengadakan](#) detail tentang mengemas file build game Anda dan membuat sumber daya build dengan file di bucket Amazon S3.
- Buat armada yang dikelola. Buat armada menggunakan konsol atau AWS CLI, menentukan armada yang EC2 dikelola. Jenis armada ini memerlukan pengaturan konfigurasi tambahan, termasuk menentukan sumber daya build dan tipe instance. Anda dapat menggunakan konfigurasi runtime yang sama untuk mengelola siklus hidup server game pada setiap komputasi armada.

Lihat [Buat EC2 armada Amazon GameLift Servers terkelola](#) detail tentang membuat armada terkelola.

- Alias armada pengalihan (opsional). Jika Anda mengatur alias untuk digunakan dengan armada Anywhere Anda, Anda dapat menggunakan kembali alias yang sama untuk armada terkelola Anda. Lihat [Buat sebuah Amazon GameLift Servers alias](#) detail tentang membuat atau memperbarui alias.

Siapkan pengujian lokal dengan Amazon GameLift Servers Anywhere

Note

Topik ini mencakup pengujian lokal untuk game yang terintegrasi dengan SDK server untuk Amazon GameLift Servers versi 5.x. Jika game Anda menggunakan server SDK versi 4.x atau yang lebih lama, lihat. [Uji integrasi Anda menggunakan Amazon GameLift Servers Lokal:](#)

Gunakan armada Amazon GameLift Servers Anywhere dan perangkat keras Anda sendiri untuk membangun dan menguji komponen game Anda secara berulang di lingkungan yang dihosting yang disimulasikan. Siapkan armada Anywhere dan daftarkan perangkat lokal untuk membuat koneksi ke Amazon GameLift Servers layanan. Instal build server game Anda ke perangkat, mulai proses server game, dan uji fungsionalitas game sesuai kebutuhan. Anda dapat memperbarui build server game sesering yang diperlukan untuk menguji setiap iterasi build baru.

Dengan armada Anywhere, Anda dapat menguji menggunakan AWS CLI atau dengan skrip pengujian. Jika Anda telah mengintegrasikan klien gameAmazon GameLift Servers, Anda dapat menjalankan klien di perangkat lokal yang sama atau di perangkat lain.

Menguji secara lokal dengan armada Anywhere sangat berguna untuk menguji integrasi server game Anda. Amazon GameLift Servers Anda memiliki visibilitas penuh ke semua aktivitas hosting di mesin lokal, serta peristiwa dan data logging.

Note

Apakah Anda menggunakan Amazon GameLift Servers plugin untuk Unreal Engine atau Unity? Alat-alat ini mencakup alur kerja terpandu untuk menyiapkan pengujian lokal dengan armada Anywhere. Ikuti dokumentasi untuk [Plugin untuk Unity: Siapkan pengujian lokal](#)

[dengan Amazon GameLift Servers Anywhere](#) atau [Plugin untuk Unreal: Tuan rumah game Anda secara lokal dengan Anywhere Amazon GameLift Servers](#).

Topik

- [Siapkan armada Anywhere lokal](#)
- [Perbarui dan instal server game Anda](#)
- [Uji aktivitas sesi permainan](#)
- [Iterasi di server game Anda](#)
- [Transisi game Anda ke armada Amazon GameLift Servers terkelola](#)

Siapkan armada Anywhere lokal

Ikuti langkah-langkah ini untuk membuat armada Anywhere untuk workstation lokal Anda. Untuk petunjuk terperinci menggunakan AWS CLI atau AWS Management Console for Amazon GameLift Servers, lihat [Buat sebuah Amazon GameLift Servers Armada di mana saja](#)

Untuk membuat armada Anywhere

1. Buat lokasi khusus untuk workstation lokal Anda. (AWS CLI atau konsol). Lokasi khusus hanyalah label untuk sumber daya komputasi yang Anda rencanakan untuk disertakan dalam armada Anywhere Anda. Nama lokasi khusus harus dimulai dengan `custom-`. Sebagai contoh: `custom-my_laptop`. Lihat [Buat lokasi kustom](#).
2. Buat armada Anywhere (AWS CLI atau konsol). Pada langkah ini, buat sumber daya armada dengan lokasi khusus untuk workstation lokal Anda. Lihat [Buat armada Anywhere](#).

Catat ID armada baru atau nilai ARN. Anda akan membutuhkan nilai ini untuk langkah selanjutnya.

3. Daftarkan workstation lokal Anda sebagai komputasi armada (hanya AWS CLI). Armada Anywhere harus memiliki setidaknya satu sumber daya komputasi untuk meng-host server game Anda. Lihat [Tambahkan komputasi ke armada](#). Untuk menambahkan komputasi ke armada, Anda memerlukan informasi berikut:
 - Nama komputasi. Setiap komputasi dalam armada harus memiliki nama yang unik.
 - Pengidentifikasi armada Anywhere. Anda dapat menggunakan salah satu `FleetID` atau `FleetArn`.

- Info koneksi komputasi. Tentukan salah satu dari `IpAddress` atau `DnsName`. Ini adalah bagaimana Amazon GameLift Servers dan klien game akan terhubung ke server game.
- Lokasi khusus di armada Anywhere.

Catat nilai `GameLiftServiceSdkEndpoint` pengembalian. Anda akan membutuhkan nilai ini saat memperbarui server game agar berjalan di armada Anywhere.

Perbarui dan instal server game Anda

Tugas ini mengasumsikan bahwa Anda telah mengintegrasikan build server game dengan Amazon GameLift Servers server SDK 5.x. Proses integrasi melibatkan penambahan kode ke server game Anda sehingga dapat berinteraksi dengan Amazon GameLift Servers layanan untuk memulai dan mengelola sesi permainan.

Untuk armada Anywhere, Anda perlu mengonfigurasi pengaturan server game tertentu secara manual. Pada armada yang Amazon GameLift Servers dikelola, pengaturan ini dikonfigurasi secara otomatis.

Untuk mempersiapkan server game Anda untuk armada Anywhere

1. Dapatkan token otentikasi. Server game Anda harus menyertakan token otentikasi dengan setiap komunikasi dengan Amazon GameLift Servers layanan. Amazon GameLift Server token autentikasi berumur pendek dan harus disegarkan secara teratur.

Sebagai praktik terbaik, buat skrip untuk menyelesaikan tugas-tugas berikut:

- Panggil tindakan AWS CLI. `get-compute-auth-token`
- Simpan nilai token yang dikembalikan di mana proses server game dapat mengambilnya, seperti dalam variabel lingkungan pada komputasi lokal.

Instal skrip dengan server game Anda di komputasi. Atur skrip untuk dijalankan sebelum memulai proses server game pertama. Saat proses server game aktif, jalankan skrip secara teratur untuk mempertahankan token autentikasi yang valid. Semua proses server game pada komputasi dapat menggunakan token autentikasi yang sama.

2. Perbarui kode server Amazon GameLift Servers game Anda. Ketika Anda mengintegrasikan kode server game Anda dengan SDK server untuk Amazon GameLift Servers, Anda menambahkan panggilan ke tindakan `InitSdk()`. Ketika server game berjalan pada

armada Anywhere, panggilan ini memerlukan parameter server tambahan. Untuk informasi selengkapnya, lihat [Inisialisasi proses server](#) dan [Server SDK 5.x untuk Amazon GameLift Servers](#) untuk bahasa pengembangan Anda. Parameter server adalah:

- `webSocketUrl`— Tetapkan parameter ini ke `GameLiftServiceSdkEndpoint` nilai yang dikembalikan saat Anda mendaftarkan komputasi dengan armada.
- `hostId`— Setel parameter ini ke nama komputasi yang Anda tentukan saat Anda mendaftarkan komputasi dengan armada Anywhere.
- `fleetId`— Atur parameter ini ke ID armada Anywhere.
- `authToken`— Setel parameter ini ke token yang dikembalikan sebagai tanggapan atas permintaan untuk mengambil token otentikaiton untuk komputasi.
- `processId`— Tetapkan parameter ini untuk mengidentifikasi proses server game yang berjalan pada komputasi lokal. Setiap proses server game bersamaan harus memiliki ID proses yang unik.

Nilai parameter server yang digunakan setiap proses server game harus spesifik untuk komputasi armada Anywhere tempat proses berjalan. Untuk detail tentang cara mendapatkan nilai yang sesuai untuk komputasi, lihat [Tambahkan komputasi ke armada](#). Sebagai praktik terbaik, atur `webSocketUrl`, `hostId`, `fleetId`, dan `authToken` sebagai variabel lingkungan pada komputasi lokal. Semua proses server yang berjalan pada komputasi akan menggunakan nilai-nilai ini.

3. Instal server game yang dibangun di komputasi lokal. Sertakan semua dependensi yang diperlukan untuk menjalankan server game.
4. Mulai satu atau beberapa proses server game yang berjalan pada komputasi lokal. Saat proses server game memanggil aksi SDK `serverProcessReady()`, prosesnya siap untuk meng-host sesi game.

Uji aktivitas sesi permainan

Uji integrasi server game Anda dengan bekerja dengan sesi permainan. Jika Anda tidak memiliki klien game yang terintegrasi dengan Amazon GameLift Servers fungsionalitas, Anda dapat menggunakan AWS CLI untuk memulai sesi permainan. Coba skenario berikut:

- Buat sesi permainan. [create-game-session](#) Perintah panggilan (atau operasi [CreateGameSession](#) API). Tentukan ID armada Anywhere dan lokasi kustom Anda. Panggilan ini mengembalikan pengenal unik untuk sesi permainan baru.

- Periksa status sesi permainan. Panggil [describe-game-sessions](#) perintah (atau tindakan [DescribeGameSessions](#) API). Tentukan ID sesi permainan. Panggilan ini mengembalikan informasi sesi permainan terperinci, termasuk status sesi permainan. Sesi permainan dalam status Aktif siap untuk pemain terhubung. Untuk mendapatkan daftar semua sesi game untuk armada, panggil [list-game-sessions](#) perintah (atau aksi [ListGameSessions](#) API).
- Connect ke sesi permainan. Jika klien game Anda memiliki kemampuan untuk bergabung dengan sesi permainan, gunakan informasi koneksi yang termasuk dalam informasi sesi permainan.

Iterasi di server game Anda

Anda dapat menggunakan Anywhere armada dan komputasi yang sama untuk menguji versi lain dari build server game Anda.

1. Bersihkan yang sudah ada **GameSession**. Jika proses server game mogok atau tidak mau menelepon `ProcessEnding()`, Amazon GameLift Servers bersihkan `GameSession` setelah server game berhenti mengirim pemeriksaan kesehatan.
2. Hasilkan build server game baru. Buat perubahan pada server game Anda dan paketkan build yang direvisi.
3. Perbarui build server game di komputasi lokal Anda. Armada Anywhere Anda sebelumnya masih aktif dan laptop Anda masih terdaftar sebagai sumber daya komputasi di armada.
4. Dapatkan token otorisasi yang diperbarui. Panggil perintah [get-compute-auth-token](#) CLI dan simpan token pada komputasi lokal.
5. Mulai satu atau beberapa proses server game yang berjalan pada komputasi lokal. Ketika server game memproses panggilan `ProcessReady()`, itu siap digunakan untuk pengujian.

Transisi game Anda ke armada Amazon GameLift Servers terkelola

Setelah Anda menyelesaikan pengujian pengembangan dan Anda siap untuk mempersiapkan peluncuran, ini adalah saat yang tepat untuk beralih ke armada Amazon GameLift Servers terkelola. Gunakan armada terkelola untuk menyempurnakan dan menguji sumber daya hosting game Anda. Terapkan solusi penempatan sesi permainan Anda (antrian dan mak comblang), pilih perangkat keras hosting optimal (termasuk armada Spot) dan lokasi, dan pilih strategi untuk kapasitas penskalaan. Anda mungkin juga ingin mulai menggunakan AWS CloudFormation untuk mengelola siklus hidup semua sumber daya hosting game Anda dengan lebih efisien, termasuk armada, antrian, dan mak comblang.

Anda perlu membuat beberapa modifikasi kecil untuk transisi dari armada uji Anywhere lokal ke armada Amazon GameLift Servers terkelola. Anda dapat menggunakan kembali antrian dan mak comblang yang sama. Lakukan tugas-tugas berikut:

- Ubah panggilan kode server game ke **InitSdk()**. Hapus parameter server. Untuk armada yang dikelola, Amazon GameLift Servers secara otomatis melacak informasi ini.
- Buat sumber daya Amazon GameLift Servers build. Dengan armada pengujian Anywhere, Anda harus menerapkan build dan dependensi server game secara manual ke setiap komputasi armada. Dengan armada terkelola, Anda membuat dan mengunggah paket build game Anda ke Amazon GameLift Servers, yang secara otomatis menerapkannya ke semua komputasi armada. Lihat [Menerapkan build server khusus untuk Amazon GameLift Servers mengadakan](#) detail tentang mengemas file build game Anda dan membuat sumber daya build dengan file di bucket Amazon S3. Jangan sertakan skrip yang mendaftarkan komputasi dan mendapatkan token otentikasi, karena Amazon GameLift Servers secara otomatis menangani tugas ini dengan armada terkelola.
- Buat armada yang dikelola. Buat armada menggunakan konsol atau AWS CLI, menentukan armada yang EC2 dikelola. Jenis armada ini memerlukan pengaturan konfigurasi tambahan, termasuk menentukan sumber daya build dan tipe instance. Anda semua perlu menyiapkan konfigurasi runtime untuk mengelola siklus hidup server game pada setiap komputasi armada. Lihat [Buat EC2 armada Amazon GameLift Servers terkelola](#) detail tentang membuat armada terkelola.
- Alias armada pengalihan (opsional). Jika Anda mengatur alias untuk digunakan dengan armada Anywhere Anda, Anda dapat menggunakan kembali alias yang sama untuk armada terkelola Anda. Lihat [Buat sebuah Amazon GameLift Servers alias](#) detail tentang membuat atau memperbarui alias.

Bekerja dengan Amazon GameLift Servers Agen

Amazon GameLift Servers Agen mengawasi jalannya proses server game di Amazon GameLift Servers armada Anda. Agen digunakan untuk setiap komputasi dalam armada, dan menyediakan manajemen proses otomatis, manajemen hosting, dan logging untuk komputasi. Untuk menggunakan Agen, Anda harus memiliki build server game yang terintegrasi dengan SDK server untuk Amazon GameLift Servers versi 5.x atau yang lebih baru.

Amazon GameLift Servers Agen tersedia secara eksternal untuk digunakan dengan Amazon GameLift Servers armada yang tidak dikelola armada. EC2 (EC2 Armada terkelola menangani tugas Agen secara otomatis.) Anda dapat memilih untuk menjalankan Amazon GameLift Servers armada,

termasuk armada Anywhere, dengan atau tanpa Agen. Tanpa Agen, Anda harus memberikan solusi alternatif untuk menyelesaikan tugas yang diperlukan.

Saat diterapkan ke komputasi, Amazon GameLift Servers Agen harus diluncurkan sebelum proses server game dimulai. Saat diluncurkan, Agen menyelesaikan tugas-tugas berikut:

- Mendaftarkan komputasi dengan armada Amazon GameLift Servers Anywhere menggunakan API. [RegisterCompute](#)
- Memanggil [GetComputeAuthToken](#) API untuk mengambil token otorisasi dan menyimpannya untuk digunakan oleh proses server yang berjalan pada komputasi.
- Menetapkan variabel lingkungan WebSocket URL untuk komputasi, dan membuat WebSocket koneksi ke layanan. Amazon GameLift Servers
- Meminta versi terbaru dari konfigurasi runtime armada dari Amazon GameLift Servers layanan.
- Memulai dan menghentikan proses server sesuai dengan instruksi konfigurasi runtime.

Kode sumber dan instruksi pembuatan untuk Amazon GameLift Servers Agen tersedia di [Amazon GameLift ServersAgen](#) GitHub.

Tentang Agen

Amazon GameLift ServersAgen dirancang untuk menangani tugas-tugas berikut untuk armada Anda:

Manajemen proses

- Memulai proses server baru seperti yang didefinisikan dalam instruksi runtime. Agen mungkin menggunakan konfigurasi runtime kustom yang diterapkan dengan Agen. Atau, Anda dapat memberikan `RuntimeConfiguration` sebagai bagian dari definisi armada Anda. Pendekatan ini memiliki keuntungan karena Anda dapat memodifikasi konfigurasi runtime armada kapan saja. Agen secara berkala meminta konfigurasi runtime yang diperbarui dari layanan. Amazon GameLift Servers
- Memantau aktivasi proses server dan menghentikan proses ketika mereka tidak aktif tepat waktu.
- Mengirim detak jantung ke. Amazon GameLift Servers Jika Agen gagal mengirim detak jantung, perhitungan mungkin ditandai sebagai basi.
- Melaporkan Amazon GameLift Servers kapan proses server berakhir. Amazon GameLift Serversmenggunakan informasi ini untuk memantau ketersediaan server game untuk penempatan sesi game.
- Memancarkan peristiwa armada untuk proses server, termasuk:

- `SERVER_PROCESS_INVALID_PATH`: Parameter peluncuran proses server game tidak dikonfigurasi dengan benar.
- `SERVER_PROCESS_TERMINATED_UNHEALTHY`: Proses server game tidak melaporkan pemeriksaan kesehatan yang valid dalam waktu 3 menit setelah pengaktifan dan oleh karena itu dihentikan.
- `SERVER_PROCESS_FORCE_TERMINATED`: Proses server game tidak keluar dengan bersih setelah `OnProcessTerminate()` dikirim dalam waktu 30 detik.
- `SERVER_PROCESS_CRASHED`: Proses server game macet karena beberapa alasan.

Manajemen komputasi

- Menerima pesan dari Amazon GameLift Servers layanan untuk mematikan komputasi.
- Meminta komputasi untuk dihentikan oleh. Amazon GameLift Servers

Pencatatan log

- Mengunggah log ke bucket Amazon S3 di AWS akun Anda.

Uji integrasi Anda menggunakan Amazon GameLift Servers Lokal:

Note

Topik ini mencakup pengujian untuk game yang terintegrasi dengan SDK server untuk Amazon GameLift Servers versi 4.x atau sebelumnya saja. Paket SDK server Anda menyertakan versi yang kompatibel Amazon GameLift Servers Lokal. Jika Anda menggunakan SDK server versi 5.x, lihat [Siapkan pengujian lokal dengan Amazon GameLift Servers Anywhere](#) pengujian lokal dengan Amazon GameLift Servers Armada di mana saja.

Gunakan Amazon GameLift Servers Lokal untuk menjalankan versi terbatas dari yang dikelola Amazon GameLift Servers layanan pada perangkat lokal dan uji integrasi game Anda terhadapnya. Alat ini berguna ketika melakukan pengembangan berulang pada integrasi game Anda. Alternatif—mengunggah setiap build baru ke Amazon GameLift Servers dan mengonfigurasi armada untuk menjadi tuan rumah game Anda—dapat memakan waktu beberapa atau lebih setiap kali.

Dengan Amazon GameLift Servers Lokal, Anda dapat memverifikasi hal berikut:

- Server game Anda terintegrasi dengan benar dengan Server SDK dan berkomunikasi dengan benar Amazon GameLift Servers layanan untuk memulai sesi permainan baru, menerima pemain baru, dan melaporkan kesehatan dan status.
- Klien game Anda terintegrasi dengan benar dengan AWS SDK untuk Amazon GameLift Servers dan dapat mengambil informasi tentang sesi permainan yang ada, memulai sesi permainan baru, bergabung dengan pemain ke game dan terhubung ke sesi permainan.

Amazon GameLift Servers Lokal adalah alat baris perintah yang memulai versi mandiri dari yang dikelola Amazon GameLift Servers layanan. Amazon GameLift Servers Local juga menyediakan log peristiwa yang berjalan dari inisialisasi proses server, pemeriksaan kesehatan, dan panggilan dan respons API. Amazon GameLift Servers Local mengenali subset tindakan AWS SDK untuk Amazon GameLift Servers. Anda dapat melakukan panggilan dari AWS CLI atau dari klien game Anda. Semua tindakan API dilakukan secara lokal seperti yang mereka lakukan di Amazon GameLift Servers layanan web.

Setiap proses server seharusnya hanya meng-host satu sesi permainan. Sesi permainan adalah executable yang Anda gunakan untuk terhubung Amazon GameLift Servers Lokal. Ketika sesi permainan selesai, Anda harus menelepon `GameLiftServerSDK::ProcessEnding` dan kemudian keluar dari proses. Saat menguji secara lokal dengan Amazon GameLift Servers Lokal, Anda dapat memulai beberapa proses server. Setiap proses akan terhubung ke Amazon GameLift Servers Lokal. Anda kemudian dapat membuat satu sesi permainan untuk setiap proses server. Ketika sesi permainan Anda berakhir, proses server game Anda harus keluar. Anda kemudian harus secara manual memulai proses server lain.

Amazon GameLift Servers lokal mendukung yang berikut APIs:

- `CreateGameSession`
- `CreatePlayerSession`
- `CreatePlayerSessions`
- `DescribeGameSessions`
- `DescribePlayerSessions`

Penyiapan Amazon GameLift Servers lokal

Amazon GameLift Servers [Lokal disediakan sebagai .jar file yang dapat dieksekusi yang dibundel dengan Server SDK](#). Ini dapat dijalankan di Windows atau Linux dan digunakan dengan apa saja Amazon GameLift Servers-bahasa yang didukung.

Sebelum menjalankan Local, Anda juga harus memiliki berikut yang diinstal.

- Sebuah build SDK server untuk Amazon GameLift Servers versi 3.1.5 hingga 4.x.
- Java 8

Uji server game

Jika Anda ingin menguji server game Anda saja, Anda dapat menggunakan AWS CLI untuk mensimulasikan panggilan klien game ke Amazon GameLift Servers Layanan lokal. Ini memverifikasi bahwa server game Anda melakukan seperti yang diharapkan dengan berikut ini:

- Server game diluncurkan dengan benar dan menginisialisasi SDK server untuk Amazon GameLift Servers.
- Sebagai bagian dari proses peluncuran, server game memberi tahu Amazon GameLift Servers bahwa server siap untuk menyelenggarakan sesi permainan.
- Server game mengirimkan status kesehatan ke Amazon GameLift Servers setiap menit saat berlari.
- Server game menanggapi permintaan untuk memulai sesi game baru.

1. Mulai Amazon GameLift Servers Lokal.

Buka jendela command prompt, arahkan ke direktori yang berisi file *GameLiftLocal.jar* dan menjalankannya. Secara default, Local mendengarkan permintaan dari klien game pada port 8080. Untuk menentukan nomor port yang berbeda, gunakan parameter `-p`, seperti yang ditunjukkan dalam contoh berikut:

```
java -jar GameLiftLocal.jar -p 9080
```

Setelah Local dimulai, Anda melihat log yang menunjukkan bahwa dua server Local dimulai, satu mendengarkan untuk server game Anda dan satu mendengarkan untuk klien game Anda atau

AWS CLI. Log terus melaporkan aktivitas pada dua server Local, termasuk komunikasi dan dari komponen game Anda.

2. Mulai server game Anda.

Mulai Anda Amazon GameLift Servers-server game terintegrasi secara lokal. Anda tidak perlu mengubah titik akhir untuk server game.

Di jendela prompt perintah Lokal, pesan log menunjukkan bahwa server game Anda telah terhubung ke Amazon GameLift Servers Layanan lokal. Ini berarti server game Anda berhasil menginisialisasi SDK server untuk Amazon GameLift Servers (dengan `InitSDK()`). Ini telah memanggil `ProcessReady()` dengan jalur log yang ditampilkan dan, jika berhasil, siap untuk meng-host sesi game. Saat server game sedang berjalan, Amazon GameLift Servers mencatat setiap laporan status kesehatan dari server game. Contoh log olah pesan berikut menunjukkan server game berhasil terintegrasi:

```
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
```

Kemungkinan kesalahan dan pesan peringatan meliputi berikut ini:

- Kesalahan: "ProcessReady tidak menemukan proses dengan PID:<process ID>! Apakah `initSDK()` dipanggil?"
- Peringatan: "Status proses sudah ada untuk proses dengan PID:<process ID>! Apakah `ProcessReady(...)` dipanggil lebih dari sekali?"

3. Mulai AWS CLI.

Setelah server game Anda berhasil memanggil `ProcessReady()`, Anda dapat mulai melakukan panggilan klien. Buka jendela prompt perintah lain dan mulai AWS CLI alat. Secara AWS CLI default menggunakan Amazon GameLift Servers titik akhir layanan web. Anda harus mengganti ini dengan titik akhir Local di setiap permintaan menggunakan parameter `--endpoint-url`, seperti yang ditunjukkan dalam permintaan contoh berikut.

```
AWS gamelift describe-game-sessions --endpoint-url http://localhost:9080 --fleet-id fleet-123
```

Di jendela AWS CLI command prompt, `AWS gamelift` perintah menghasilkan respons seperti yang didokumentasikan dalam [AWS CLI Command Reference](#).

4. Buat sesi permainan.

Dengan AWS CLI, kirimkan permintaan [CreateGameSession\(\)](#). Permintaan harus mengikuti sintaks yang diharapkan. Untuk Local, parameter `FleetId` dapat diatur ke string yang valid()`^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
```

Di jendela prompt perintah Lokal, pesan log menunjukkan bahwa Amazon GameLift Servers Lokal telah mengirim `onStartGameSession` panggilan balik server game Anda. Jika sesi game berhasil dibuat, server game Anda merespon dengan menerapkan `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: gameSessionId:
"arn:aws:gamelift:local::gamesession/
fleet-1a2b3c4d-5e6f-7a8b-9c0d-1e2f3a4b5c6d/gsess-abcdef12-3456-7890-abcd-
ef1234567890"
```

Di AWS CLI jendela, Amazon GameLift Servers merespons dengan objek sesi permainan termasuk ID sesi game. Perhatikan bahwa status sesi game baru adalah Mengaktifkan. Status berubah menjadi Aktif setelah server game Anda dipanggil. `ActivateGameSession` Jika Anda ingin melihat status yang diubah, gunakan tombol AWS CLI `to call DescribeGameSessions()`.

```
{
  "GameSession": {
    "Status": "ACTIVATING",
    "MaximumPlayerSessionCount": 2,
    "FleetId": "fleet-1a2b3c4d-5e6f-7a8b-9c0d-1e2f3a4b5c6d",
    "GameSessionId": "arn:aws:gamelift:local::gamesession/
fleet-1a2b3c4d-5e6f-7a8b-9c0d-1e2f3a4b5c6d/gsess-abcdef12-3456-7890-abcd-
ef1234567890",
    "IpAddress": "127.0.0.1",
    "Port": 1935
  }
}
```

Uji server game dan klien

Untuk memeriksa integrasi game lengkap Anda, termasuk menghubungkan pemain ke game, Anda dapat menjalankan server game dan klien secara Local. Ini memungkinkan Anda untuk menguji panggilan terprogram dari klien game Anda ke Amazon GameLift Servers Lokal. Anda dapat memverifikasi tindakan berikut:

- Klien game berhasil membuat permintaan AWS SDK ke Amazon GameLift Servers Layanan lokal, termasuk untuk membuat sesi permainan, mengambil informasi tentang sesi permainan yang ada, dan membuat sesi pemain.
- Server game betul memvalidasi pemain ketika mereka mencoba untuk bergabung dengan sesi game. Untuk pemain divalidasi, server game dapat mengambil data pemain (jika diimplementasikan).
- Server game melaporkan connection terputus ketika pemain meninggalkan game.

- Laporan server game mengakhiri sesi game.

1. Mulai Amazon GameLift Servers Lokal.

Buka jendela command prompt, arahkan ke direktori yang berisi file *GameLiftLocal.jar* dan menjalankannya. Secara default, Local mendengarkan permintaan dari klien game pada port 8080. Untuk menentukan nomor port yang berbeda, gunakan parameter `-p`, seperti yang ditunjukkan dalam contoh berikut.

```
./gamelift-local -p 9080
```

Setelah Local dimulai, Anda melihat log yang menunjukkan bahwa dua server Local dimulai, satu mendengarkan server game Anda dan satu mendengarkan untuk klien game Anda atau AWS CLI.

2. Mulai server game Anda.

Mulai Anda Amazon GameLift Servers-server game terintegrasi secara lokal. Lihat [Uji server game](#) untuk detail lebih lanjut tentang log pesan.

3. Konfigurasi klien game Anda untuk Lokal dan mulai.

Untuk menggunakan klien game Anda dengan Amazon GameLift Servers Layanan lokal, Anda harus membuat perubahan berikut pada pengaturan klien game Anda, seperti yang dijelaskan dalam [Siapkan Amazon GameLift Servers pada layanan backend](#):

- Mengubah `ClientConfiguration` objek untuk menunjuk ke titik akhir Local Anda, seperti `http://localhost:9080`.
- Atur nilai ID armada target. Untuk Local, Anda tidak memerlukan ID armada nyata; atur armada target ke string yang valid (`^fleet-\S+`), seperti `fleet-1a2b3c4d-5e6f-7a8b-9c0d-1e2f3a4b5c6d`.
- Tetapkan AWS kredensialnya. Untuk Local, Anda tidak perlu kredensial nyata AWS ; Anda dapat mengatur access key dan kunci rahasia ke string apapun.

Di jendela prompt perintah Lokal, setelah Anda memulai klien game, pesan log harus menunjukkan bahwa ia telah menginisialisasi `GameLiftClient` dan berhasil dikomunikasikan dengan Amazon GameLift Servers layanan.

4. Uji panggilan klien game ke Amazon GameLift Servers layanan.

Verifikasi bahwa klien game Anda berhasil membuat salah satu atau semua panggilan API berikut:

- [CreateGameSession\(\)](#)
- [DescribeGameSessions\(\)](#)
- [CreatePlayerSession\(\)](#)
- [CreatePlayerSessions\(\)](#)
- [DescribePlayerSessions\(\)](#)

Di jendela prompt perintah Local, hanya panggilan ke `CreateGameSession()` yang menghasilkan log olah pesan. Pesan log ditampilkan kapan Amazon GameLift Servers Lokal meminta server game Anda untuk memulai sesi permainan (`onStartGameSessioncallback`) dan berhasil `ActivateGameSession` ketika server game Anda memanggilnya. Di jendela AWS CLI, semua panggilan API menghasilkan respons atau pesan kesalahan seperti yang didokumentasikan.

5. Verifikasi bahwa server game Anda memvalidasi koneksi pemain baru.

Setelah membuat sesi game dan sesi pemain, buat connection langsung ke sesi game.

Di jendela prompt perintah Local, log olah pesan harus menunjukkan bahwa server game telah mengirim `AcceptPlayerSession()` untuk memvalidasi connection pemain baru. Jika Anda menggunakan AWS CLI to call `DescribePlayerSessions()`, status sesi pemain harus berubah dari `Reserved` ke `Active`.

6. Verifikasi bahwa server game Anda melaporkan status game dan pemain ke Amazon GameLift Servers layanan.

Untuk Amazon GameLift Servers untuk mengelola permintaan pemain dan melaporkan metrik dengan benar, server game Anda harus melaporkan berbagai status kembali Amazon GameLift Servers. Verifikasi bahwa Lokal mencatat peristiwa yang terkait dengan tindakan berikut. Anda mungkin juga ingin menggunakan AWS CLI untuk melacak perubahan status.

- Pemain terputus dari sesi permainan — Amazon GameLift Servers Pesan log lokal harus menunjukkan bahwa server game Anda memanggil `RemovePlayerSession()`. Sebuah AWS CLI panggilan ke `DescribePlayerSessions()` harus mencerminkan perubahan status dari `Active` ke `Completed`. Anda juga dapat menghubungi `DescribeGameSessions()` untuk memeriksa bahwa jumlah pemain sesi game saat ini berkurang satu per satu.

- Sesi permainan berakhir — Amazon GameLift Servers Pesan log lokal harus menunjukkan bahwa server game Anda memanggil `TerminateGameSession()`.

 Note

Bimbingan sebelumnya adalah memanggil `TerminateGameSession()` saat mengakhiri sesi game. Metode ini tidak digunakan lagi dengan Amazon GameLift Servers Server SDK v4.0.1. Lihat [Mengakhiri sesi game](#).

- Proses server dihentikan - Amazon GameLift Servers Pesan log lokal harus menunjukkan bahwa server game Anda memanggil `ProcessEnding()`. AWS CLI Panggilan ke `DescribeGameSessions()` harus mencerminkan perubahan status dari `Active` ke `Terminated` (atau `Terminating`).

Variasi dengan lokal

Saat menggunakan Amazon GameLift Servers Lokal, perlu diingat hal-hal berikut:

- Berbeda dengan Amazon GameLift Servers layanan web, Lokal tidak melacak status kesehatan server dan memulai `onProcessTerminate` panggilan balik. Local hanya berhenti mencatat laporan kondisi untuk server game.
- Untuk panggilan ke AWS SDK, armada tidak IDs divalidasi, dan dapat berupa nilai string apa pun yang memenuhi persyaratan parameter `()^fleet-\S+`.
- Sesi permainan IDs yang dibuat dengan Lokal memiliki struktur yang berbeda. Mereka termasuk string `local`, seperti yang ditunjukkan di sini:

```
arn:aws:gamelift:local::gamesession/fleet-123/gsess-56961f8e-  
db9c-4173-97e7-270b82f0daa6
```

Menambahkan FlexMatch perjodohan

Gunakan Amazon GameLift Servers FlexMatch untuk menambahkan fungsionalitas perjodohan pemain ke game yang Anda Amazon GameLift Servers host. Anda dapat menggunakan FlexMatch dengan server game khusus atau Amazon GameLift ServersRealtime.

FlexMatchmemasangkan layanan perjodohan dengan mesin aturan yang dapat disesuaikan. Anda merancang cara mencocokkan pemain berdasarkan atribut pemain dan mode permainan yang masuk akal untuk permainan Anda. FlexMatchmengelola mur dan baut mengevaluasi pemain yang mencari permainan, membentuk pertandingan dengan satu atau lebih tim, dan memulai sesi permainan untuk menjadi tuan rumah pertandingan.

Untuk menggunakan FlexMatch layanan penuh, Anda harus menyiapkan sumber daya hosting Anda dengan antrian. Amazon GameLift Serversmenggunakan antrian untuk menemukan lokasi hosting terbaik untuk game di berbagai wilayah dan jenis komputasi. Secara khusus, Amazon GameLift Servers antrian dapat menggunakan data latensi, ketika disediakan oleh klien game, untuk menempatkan sesi permainan sehingga pemain mengalami latensi serendah mungkin saat bermain.

Untuk informasi selengkapnya tentang FlexMatch menyertakan bantuan mendetail dalam mengintegrasikan perjodohan ke dalam game Anda, lihat topik Panduan [Amazon GameLift ServersFlexMatchPengembang ini](#):

- [Bagaimana cara Amazon GameLift ServersFlexMatch kerja](#)
- [FlexMatchlangkah-langkah integrasi](#)

Dapatkan data armada untuk sebuah Amazon GameLift Servers instance

Ada beberapa situasi di mana pembuatan atau Amazon GameLift Servers Realtime skrip game khusus Anda mungkin memerlukan informasi tentang Amazon GameLift Servers armada. Misalnya, bangunan atau skrip game Anda mungkin menyertakan kode untuk:

- Memantau aktivitas berdasarkan data armada.
- Gulung metrik untuk melacak aktivitas berdasarkan data armada. (Banyak game menggunakan data ini untuk LiveOps kegiatan.)
- Berikan data yang relevan ke layanan game khusus, seperti untuk perjodohan, penskalaan kapasitas tambahan, atau pengujian.

Informasi armada tersedia sebagai file JSON pada setiap instans di lokasi-lokasi berikut:

- Windows: C:\GameMetadata\gamelift-metadata.json
- Linux: /local/gamemetadata/gamelift-metadata.json

gamelift-metadata.jsonFile tersebut mencakup [atribut sumber daya Amazon GameLift Servers armada](#).

Contoh file 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"
}
```

Mengintegrasikan game dengan Amazon GameLift ServersRealtime

Topik ini memberikan gambaran umum tentang Amazon GameLift Servers Realtime solusi yang dikelola Amazon GameLift Servers dengan. Ikhtisar menjelaskan kapan solusi ini cocok untuk game Anda, dan bagaimana Amazon GameLift Servers Realtime mendukung game multipemain.

Tip

[Mulai sekarang dengan Amazon GameLift Servers](#)

Apa itu Realtime server?

Realtimeserver ringan, server ready-to-go game yang Amazon GameLift Servers menyediakan untuk Anda gunakan dengan game multipemain Anda. Realtimeserver menghapus proses pengembangan, pengujian, dan penyebaran server game kustom. Solusi ini dapat membantu meminimalkan waktu dan upaya yang diperlukan untuk menyelesaikan permainan Anda.

Fitur utama

- Tumpukan jaringan penuh untuk klien game dan interaksi server
- Fungsionalitas server game inti
- Logika server yang dapat disesuaikan
- Pembaruan langsung untuk Realtime konfigurasi dan logika server
- FlexMatchperjodohan
- Kontrol sumber daya hosting yang fleksibel

Siapkan Realtime server dengan membuat armada dan menyediakan skrip konfigurasi.

Bagaimana Amazon GameLift ServersRealtime mengelola sesi permainan

Anda dapat menambahkan logika khusus untuk manajemen sesi game dengan membangunnya ke dalam Realtime skrip. Anda dapat menulis kode untuk mengakses objek khusus server, menambahkan logika berbasis peristiwa menggunakan panggilan balik, atau menambahkan logika berdasarkan skenario non-peristiwa.

Bagaimana Realtime klien dan server berinteraksi

Selama sesi permainan, klien game berinteraksi dengan mengirim pesan ke Realtime server melalui layanan backend. Layanan backend kemudian menyampaikan pesan di antara klien game untuk bertukar aktivitas, status game, dan data game yang relevan.

Selain itu, Anda dapat menyesuaikan bagaimana klien dan server berinteraksi dengan menambahkan logika permainan ke Realtime skrip. Dengan logika permainan khusus, Realtime server mungkin menerapkan callback untuk memulai respons berbasis peristiwa.

Protokol komunikasi

Realtimeserver dan klien game yang terhubung berkomunikasi melalui dua saluran: koneksi TCP untuk pengiriman yang andal, dan saluran UDP untuk pengiriman cepat. Saat membuat pesan, client game memilih protokol mana yang akan digunakan tergantung pada sifat pesan. Pengiriman pesan diatur ke UDP secara default. Jika saluran UDP tidak tersedia, Amazon GameLift Servers kirim pesan menggunakan TCP sebagai fallback.

Konten pesan

Isi pesan terdiri dari dua elemen: kode operasi yang diperlukan (opCode) dan muatan opsional. OpCode pesan mengidentifikasi aktivitas pemain atau peristiwa permainan tertentu, dan payload menyediakan data tambahan yang terkait dengan kode operasi. Kedua elemen ini didefinisikan oleh pengembang. Klien game Anda bertindak berdasarkan OpCodes dalam pesan yang diterimanya.

Grup pemain

Amazon GameLift ServersRealtime menyediakan fungsionalitas untuk mengelola kelompok pemain. Secara default, Amazon GameLift Servers menempatkan semua pemain yang terhubung ke game dalam grup “semua pemain”. Selain itu, pengembang dapat mengatur grup lain untuk game mereka, dan pemain dapat menjadi anggota dari beberapa grup secara bersamaan. Anggota grup dapat mengirim pesan dan berbagi data game dengan semua pemain dalam grup. Satu kemungkinan penggunaan untuk grup adalah menyiapkan tim pemain dan mengelola komunikasi tim.

Amazon GameLift ServersRealtime dengan sertifikat TLS

Dengan Amazon GameLift ServersRealtime, otentikasi server dan enkripsi paket data dibangun ke dalam layanan. Fitur keamanan ini diaktifkan saat Anda mengaktifkan pembuatan sertifikat TLS. Ketika klien game mencoba terhubung dengan Realtime server, server secara otomatis merespons dengan sertifikat TLS, yang divalidasi klien. Amazon GameLift Servers menangani enkripsi menggunakan TLS untuk komunikasi TCP (WebSockets) dan DTLS untuk lalu lintas UDP.

Menyesuaikan server Realtime

RealtimeServer berfungsi sebagai server relay stateless. RealtimeServer menyampaikan paket pesan dan data game antara klien game yang terhubung ke game. Namun, Realtime server tidak mengevaluasi pesan, memproses data, atau melakukan logika gameplay apa pun. Digunakan dengan cara ini, setiap klien game mempertahankan pandangannya sendiri tentang status permainan dan memberikan pembaruan kepada pemain lain melalui server relai. Setiap client game bertanggung jawab untuk menggabungkan pembaruan ini dan menyesuaikan status gamenya sendiri.

Anda dapat menyesuaikan server Anda dengan menambahkan fungsionalitas Realtime skrip. Dengan logika game, misalnya, Anda dapat membangun game stateful dengan tampilan server-otoritatif dari status game.

Amazon GameLift Serversmendefinisikan satu set callback sisi server untuk skrip. Realtime Menerapkan callback ini untuk menambahkan fungsionalitas berdasarkan kejadian ke server Anda. Sebagai contoh, Anda dapat:

- Mengautentikasi pemain ketika client game mencoba untuk menyambung ke server.
- Validasi apakah pemain dapat bergabung dengan grup berdasarkan permintaan.
- Tentukan kapan harus mengirim pesan dari pemain tertentu atau ke pemain target, atau melakukan pemrosesan tambahan sebagai tanggapan.
- Beri tahu semua pemain saat pemain meninggalkan grup atau terputus dari server.
- Lihat konten objek sesi permainan atau objek pesan, dan gunakan data.

Menyebarkan dan memperbarui Amazon GameLift ServersRealtime

Keuntungan utama Amazon GameLift Servers Realtime adalah kemampuan untuk memperbarui skrip Anda kapan saja. Saat Anda memperbarui skrip, Amazon GameLift Servers mendistribusikan versi baru ke semua sumber daya hosting dalam beberapa menit. Setelah Amazon GameLift Servers menyebarkan skrip baru, semua sesi permainan baru yang dibuat setelah titik itu akan menggunakan versi skrip baru. (Sesi permainan yang ada akan terus menggunakan versi aslinya.)

Mulailah mengintegrasikan game Anda dengan Amazon GameLift ServersRealtime:

- [Mengintegrasikan klien game untuk Amazon GameLift ServersRealtime](#)
- [Membuat Realtime skrip](#)

Mengintegrasikan klien game untuk Amazon GameLift ServersRealtime

Topik ini menjelaskan cara mempersiapkan klien game Anda untuk dapat bergabung dan berpartisipasi dalam sesi permainan yang Amazon GameLift Servers di-host.

Ada dua set tugas yang diperlukan untuk mempersiapkan klien game Anda:

- Siapkan klien game Anda untuk memperoleh informasi tentang game yang tersedia, meminta matchmaking, memulai sesi game baru, dan menyimpan slot sesi game untuk pemain.
- Aktifkan klien game Anda untuk bergabung dengan sesi game yang dihosting di Realtime server dan bertukar pesan.

Temukan atau buat sesi permainan dan sesi pemain

Siapkan klien game Anda untuk menemukan atau memulai sesi permainan, meminta FlexMatch perjodohan, dan menyediakan ruang untuk pemain dalam game dengan membuat sesi pemain.

Sebagai praktik terbaik, buat layanan backend dan gunakan untuk membuat permintaan langsung ke Amazon GameLift Servers layanan saat dipicu oleh tindakan klien game. Layanan backend kemudian menyampaikan respons yang relevan kembali ke klien game.

1. Tambahkan AWS SDK ke klien game Anda, inisialisasi Amazon GameLift Servers klien, dan konfigurasi untuk menggunakan sumber daya hosting di armada dan antrian Anda. AWS SDK tersedia dalam beberapa bahasa; lihat. Amazon GameLift Servers SDKs [Untuk layanan klien game](#)
2. Tambahkan Amazon GameLift Servers fungsionalitas ke layanan backend Anda. Untuk petunjuk lebih rinci, lihat [Tambahkan Amazon GameLift Servers ke klien game Anda](#) dan [Menambahkan FlexMatch perjodohan](#). Praktik terbaik adalah menggunakan penempatan sesi game untuk membuat sesi game baru. Metode ini memungkinkan Anda memanfaatkan sepenuhnya Amazon GameLift Servers kemampuan untuk menempatkan sesi permainan baru dengan cepat dan cerdas, serta menggunakan data latensi pemain untuk meminimalkan jeda permainan. Minimal, layanan backend Anda harus dapat meminta sesi permainan baru dan menangani data sesi game sebagai tanggapan. Anda mungkin juga ingin menambahkan fungsionalitas untuk mencari dan mendapatkan informasi tentang sesi game yang ada, dan meminta sesi pemain, yang secara efektif menyimpan slot pemain di sesi game yang ada.
3. Menyampaikan informasi koneksi kembali ke klien game. Layanan backend menerima sesi permainan dan objek sesi pemain dalam menanggapi permintaan ke layanan. Amazon GameLift Servers Objek ini berisi informasi, khususnya detail koneksi (alamat IP dan port) dan ID sesi pemain, yang harus connect oleh klien game ke sesi game yang berjalan di Server Realtime.

Connect ke game di Amazon GameLift ServersRealtime

Aktifkan klien game Anda untuk terhubung langsung dengan sesi permainan yang dihosting di Realtime server dan bertukar pesan dengan server dan dengan pemain lain.

1. Dapatkan SDK klien Amazon GameLift ServersRealtime, buat, dan tambahkan ke proyek klien game Anda. Lihat file README Untuk informasi lebih lanjut tentang persyaratan SDK dan instruksi tentang cara membangun pustaka klien.
2. Panggilan [Client\(\)](#) dengan konfigurasi klien yang menentukan jenis client/server koneksi yang akan digunakan.

 Note

Jika Anda menyambung ke server Realtime yang berjalan di armada aman dengan sertifikat TLS, Anda harus menentukan jenis sambungan yang aman.

3. Tambahkan fungsionalitas berikut untuk klien game Anda. Untuk informasi lebih lanjut, lihat [Amazon GameLift ServersRealtimereferensi API klien \(C #\)](#).

- Connect ke dan putuskan sambungan dari game
 - [Connect\(\)](#)
 - [Disconnect\(\)](#)
- Mengirim pesan ke penerima target
 - [SendMessage\(\)](#)
- Menerima dan memproses pesan
 - [OnDataReceived\(\)](#)
- Bergabung dengan grup dan tinggalkan grup pemain
 - [JoinGroup\(\)](#)
 - [RequestGroupMembership\(\)](#)
 - [LeaveGroup\(\)](#)

4. Mengatur event handler untuk callback klien yang diperlukan. Lihat [Amazon GameLift ServersRealtimereferensi API klien \(C #\): Panggilan balik asinkron](#).

Ketika bekerja dengan armada Realtime yang memiliki generasi sertifikat TLS diaktifkan, server secara otomatis dikonfirmasi menggunakan sertifikat TLS. Lalu lintas TCP dan UDP dienkripsi dalam penerbangan untuk menyediakan keamanan lapisan pengangkutan. Lalu lintas TCP dienkripsi menggunakan TLS 1.2, dan lalu lintas UDP dienkripsi menggunakan DTLS 1.2.

Contoh klien game

Klien realtime dasar (C #)

Contoh ini menggambarkan integrasi klien game dasar dengan SDK klien (C#) untuk Amazon GameLift Servers Realtime Seperti yang ditunjukkan, contoh menginisialisasi objek Realtime klien, menyiapkan event handler dan mengimplementasikan callback sisi klien, menghubungkan ke Realtime server, mengirim pesan, dan memutuskan sambungan.

```

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);
}
}
```

Membuat Realtime skrip

Untuk digunakan Amazon GameLift Servers Realtime untuk permainan Anda, Anda perlu menyediakan skrip (dalam bentuk beberapa JavaScript kode) untuk mengkonfigurasi dan secara opsional menyesuaikan armada. Amazon GameLift Servers Realtime Topik ini mencakup langkah-langkah kunci dalam membuat Realtime skrip. Setelah skrip siap, unggah ke Amazon GameLift Servers layanan dan gunakan untuk membuat armada (lihat [Menyebarkan skrip untuk Amazon GameLift ServersRealtime](#)).

Untuk menyiapkan skrip untuk digunakan Amazon GameLift ServersRealtime, tambahkan fungsionalitas berikut ke Realtime skrip Anda.

Kelola siklus hidup sesi game (wajib)

Minimal, Realtime skrip harus menyertakan `Init()` fungsi, yang mempersiapkan Realtime server untuk memulai sesi permainan. Juga sangat disarankan agar Anda juga menyediakan cara untuk mengakhiri sesi game, untuk memastikan bahwa sesi game baru dapat terus dimulai di armada Anda.

Fungsi `Init()` callback, ketika dipanggil, dilewatkan objek Realtime sesi, yang berisi antarmuka untuk Realtime server. Lihat [Amazon GameLift ServersRealtimeantarmuka](#) untuk detail lebih lanjut tentang antarmuka ini.

Untuk mengakhiri sesi permainan dengan anggun, skrip juga harus memanggil fungsi Realtime server. `session.processEnding` Hal ini memerlukan beberapa mekanisme untuk menentukan kapan harus mengakhiri sesi. Kode contoh skrip menggambarkan mekanisme sederhana yang memeriksa koneksi pemain dan memicu penghentian sesi game ketika tidak ada pemain yang terhubung ke sesi selama jangka waktu tertentu.

Amazon GameLift ServersRealtime dengan konfigurasi paling dasar - inisialisasi dan penghentian proses server - pada dasarnya bertindak sebagai server relay stateless. RealtimeServer menyampaikan pesan dan data game antara klien game yang terhubung ke game, tetapi tidak mengambil tindakan independen untuk memproses data atau melakukan logika. Anda dapat secara opsional menambahkan logika game, yang dipicu oleh peristiwa game atau mekanisme lain, sesuai kebutuhan untuk game Anda.

Tambahkan logika permainan sisi server (opsional)

Anda dapat menambahkan logika permainan secara opsional ke Realtime skrip Anda. Misalnya, Anda dapat menjalankan perintah berikut. Kode contoh skrip memberikan ilustrasi. Lihat [Referensi skrip untuk Amazon GameLift ServersRealtime](#).

- Tambahkan logika berbasis peristiwa. Menerapkan fungsionalitas callback untuk menanggapi peristiwa client-server. Lihat [Callback skrip untuk Amazon GameLift ServersRealtime](#) untuk daftar lengkap callback.
- Memicu logika dengan mengirim pesan ke server. Buat satu set kode operasi khusus untuk pesan yang dikirim dari klien game ke server, dan tambahkan fungsionalitas untuk menangani tanda terima. Gunakan callback `onMessage`, dan kurangi konten pesan menggunakan antarmuka `gameMessage` (lihat [gameMessage.opcode](#)).
- Aktifkan logika game untuk mengakses AWS sumber daya Anda yang lain. Lihat perinciannya di [Berkomunikasi dengan AWS sumber daya lain dari armada Anda](#).
- Izinkan logika game mengakses informasi armada untuk instans yang sedang dijalankan. Lihat perinciannya di [Dapatkan data armada untuk sebuah Amazon GameLift Servers instance](#).

Amazon GameLift ServersRealtime contoh skrip

Contoh ini menggambarkan skrip dasar yang diperlukan untuk menyebarkan Amazon GameLift Servers Realtime ditambah beberapa logika khusus. Ini berisi fungsionalitas `Init()` yang diperlukan, dan menggunakan mekanisme pengatur waktu untuk memicu penghentian sesi game berdasarkan lamanya waktu tanpa koneksi pemain. Ini juga mencakup beberapa hook untuk logika kustom, termasuk beberapa implementasi callback.

```
// 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
};
```

Menyebarkan perangkat lunak server game untuk Amazon GameLift Servers hosting

Terapkan perangkat lunak server game multipemain Anda dengan menginstalnya di sumber daya hosting Anda, meluncurkan proses server game, dan menyiapkannya untuk meng-host game untuk pemain. Langkah-langkah untuk menyiapkan perangkat lunak server game untuk penyebaran tergantung pada jenis Amazon GameLift Servers solusi yang Anda gunakan. Dalam semua skenario, perangkat lunak server game yang digunakan berinteraksi dengan Amazon GameLift Servers layanan untuk menangani penempatan sesi game dan mengkomunikasikan detail koneksi untuk klien game.

Perangkat lunak server game Anda harus terlebih dahulu diintegrasikan dengan Amazon GameLift Servers, seperti yang dijelaskan dalam [Mempersiapkan game untuk Amazon GameLift Servers](#).

Topik di bagian ini memberikan panduan tentang cara menyiapkan perangkat lunak Anda untuk diterapkan ke skenario berikut.

- Perangkat lunak server game khusus
 - Untuk EC2 hosting Amazon GameLift Servers terkelola
 - Untuk Amazon GameLift Servers hosting kontainer terkelola
 - Untuk Hosting Amazon GameLift Servers Di Mana Saja
- Amazon GameLift Servers Realtime skrip yang disesuaikan untuk EC2 hosting Amazon GameLift Servers terkelola

Topik

- [Menerapkan build server khusus untuk Amazon GameLift Servers mengadakan](#)
- [Membangun gambar kontainer untuk Amazon GameLift Servers](#)
- [Menyebarkan skrip untuk Amazon GameLift Servers Realtime](#)

Menerapkan build server khusus untuk Amazon GameLift Servers mengadakan

Setelah Anda mengintegrasikan server game Anda dengan Amazon GameLift Servers (lihat [Mempersiapkan game untuk Amazon GameLift Servers](#)), instal perangkat lunak server game

ke sumber daya komputasi Anda untuk hosting. Proses ini bervariasi tergantung pada jenis Amazon GameLift Servers hosting yang Anda gunakan.

Menyebarkan untuk hosting terkelola

Jika Anda menggunakan Amazon GameLift Servers EC2 hosting terkelola, Anda harus mengemas perangkat lunak server game Anda dan mengunggahnya ke Amazon GameLift Servers. Saat Anda membuat armada terkelola, Amazon GameLift Servers secara otomatis menyebarkannya ke setiap instance armada.

Topik di bagian ini menjelaskan cara mengemas file build untuk diunggah, membuat skrip instalasi build opsional, lalu mengunggah file menggunakan [AWS Command Line Interface \(AWS CLI\)](#) atau AWS SDK.

Menyebarkan untuk hosting di mana saja

Jika Anda menggunakan Amazon GameLift Servers Di mana saja armada untuk hosting yang dikelola sendiri, Anda bertanggung jawab untuk menginstal perangkat lunak server game ke setiap komputasi dalam armada dan terus memperbaruinya.

Ketika proses server game terintegrasi mulai berjalan, secara otomatis menginisialisasi dan membangun komunikasi dengan Amazon GameLift Servers layanan. Proses server memulai sesi permainan dari petunjuk oleh Amazon GameLift Servers dan melaporkan aktivitas kembali ke layanan.

Topik

- [Membuat paket file bangunan game Anda](#)
- [Menambahkan skrip instalasi bangunan](#)
- [Buat Amazon GameLift Servers membangun sumber daya untuk hosting terkelola](#)
- [Perbarui build server game untuk Amazon GameLift Servers hosting terkelola](#)

Membuat paket file bangunan game Anda

Sebelum mengunggah server game Anda yang dikonfigurasi ke Amazon GameLift Servers, kemas file build game ke dalam direktori build. Proses ini merupakan persyaratan saat hosting dengan armada yang EC2 dikelola, dan merupakan praktik terbaik saat menjadi tuan rumah dengan armada Anywhere. Direktori build harus menyertakan semua komponen yang diperlukan untuk menjalankan server game dan menghosting sesi game Anda. Ini mungkin termasuk yang berikut:

- **Binari server game** – file biner yang diperlukan untuk menjalankan server game. Build dapat menyertakan binari untuk beberapa server game yang dibangun untuk berjalan pada platform yang sama. Untuk daftar platform yang didukung, lihat [Dapatkan alat Amazon GameLift Servers pengembangan](#).
- **Dependensi** — File dependen apa pun yang diperlukan oleh server game Anda untuk dijalankan. Contohnya termasuk aset, file konfigurasi, dan perpustakaan dependen.

Note

Untuk build game yang dibuat dengan SDK server untuk Amazon GameLift Servers untuk C++ (termasuk yang dibuat dengan plugin Unreal), sertakan OpenSSL DLL untuk versi OpenSSL yang sama dengan yang Anda buat SDK server. Lihat file SDK README server untuk detail selengkapnya.

- **Instal skrip (Opsional)** - File skrip untuk menangani tugas yang menginstal build game Anda Amazon GameLift Servers server hosting. Tempatkan file ini di root direktori build. Amazon GameLift Servers menjalankan skrip penginstalan sebagai bagian dari pembuatan armada.

Anda dapat menyiapkan aplikasi apa pun di build, termasuk skrip penginstalan, untuk mengakses sumber daya Anda dengan aman di AWS layanan lain. Untuk informasi tentang cara melakukan ini, lihat [Berkomunikasi dengan AWS sumber daya lain dari armada Anda](#).

Setelah mengemas file build Anda, pastikan server game Anda dapat berjalan pada instalasi bersih OS target Anda untuk memverifikasi bahwa semua dependensi yang diperlukan disertakan dan skrip penginstalan Anda akurat.

Menambahkan skrip instalasi bangunan

Buat skrip penginstalan untuk sistem operasi (OS) build game Anda:

- **Windows:** Buat file batch bernama `install.bat`.
- **Linux:** Buat file skrip shell bernama `install.sh`.

Saat membuat skrip instalasi, perhatikan hal-hal berikut:

- Skrip tidak dapat mengambil masukan pengguna apa pun.

- Amazon GameLift Servers menginstal build dan membuat ulang direktori file dalam paket build Anda di server hosting di lokasi berikut:
 - Armada Windows: C:\game
 - Armada Linux: /local/game
- Selama proses instalasi untuk armada Linux, pengguna run-as memiliki akses terbatas ke struktur file instance. Pengguna ini memiliki hak penuh atas direktori tempat file build Anda diinstal. Jika skrip penginstalan Anda melakukan tindakan yang memerlukan izin administrator, maka tentukan akses admin menggunakan sudo. Pengguna run-as untuk armada Windows memiliki izin administrator secara default. Kegagalan izin yang terkait dengan skrip instalasi menghasilkan pesan peristiwa yang menunjukkan adanya masalah dengan skrip.
- Di Linux, Amazon GameLift Servers mendukung bahasa penerjemah shell umum seperti bash. Tambahkan shebang (seperti #!/bin/bash) ke bagian atas skrip instalasi Anda. Untuk memverifikasi dukungan untuk perintah shell pilihan Anda, akses instance Linux aktif dari jarak jauh dan buka shell prompt. Untuk informasi selengkapnya, lihat [Terhubung dari jarak jauh ke Amazon GameLift Servers contoh armada](#).
- Skrip penginstalan tidak dapat mengandalkan koneksi peering VPC. Koneksi peering VPC tidak tersedia sampai setelahnya Amazon GameLift Servers menginstal build pada instance armada.

Example Windows menginstal file bash

install.bat File contoh ini menginstal komponen runtime Visual C++ yang diperlukan untuk server game dan menulis hasilnya ke file log. Skrip menyertakan file komponen dalam paket build di root.

```
vcredis_x64.exe /install /quiet /norestart /log c:\game\vcredis_2013_x64.log
```

Example Linux menginstal skrip shell

install.sh File contoh ini menggunakan bash dalam skrip instal dan menulis hasil ke file log.

```
#!/bin/bash  
echo 'Hello World' > install.log
```

install.sh File contoh ini menunjukkan bagaimana Anda dapat menggunakan CloudWatch agen Amazon untuk mengumpulkan metrik tingkat sistem dan kustom, serta menangani rotasi log. Karena Amazon GameLift Servers berjalan di VPC layanan, Anda harus memberikan Amazon GameLift Servers izin untuk mengambil peran AWS Identity and Access Management (IAM) atas nama Anda.

Untuk memungkinkan Amazon GameLift Servers untuk mengambil peran, membuat peran yang menyertakan kebijakan AWS terkelola CloudWatchAgentAdminPolicy, dan menggunakan peran tersebut saat Anda membuat armada.

```
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
```

Buat Amazon GameLift Servers membangun sumber daya untuk hosting terkelola

Saat membuat bangunan dan mengunggah file Anda, Anda memiliki beberapa pilihan:

- [Membuat bangunan dari direktori file](#). Ini adalah opsi paling sederhana dan paling umum digunakan.
- [Buat build dengan file di Amazon Simple Storage Service \(Amazon S3\)](#). Dengan opsi ini, Anda dapat mengelola versi build di Amazon S3.

Dengan kedua metode tersebut, Amazon GameLift Servers membuat sumber daya build baru dengan ID build unik dan metadata lainnya. Build dimulai dalam status Inisialisasi. Setelah Amazon GameLift Servers memperoleh file server game, build pindah ke status Ready.

Saat build sudah siap, Anda dapat menerapkannya ke yang baru Amazon GameLift Servers armada. Untuk informasi selengkapnya, lihat [Buat EC2 armada Amazon GameLift Servers terkelola](#). Kapan Amazon GameLift Servers menyiapkan armada baru, mengunduh file build ke setiap instance armada dan menginstal file build.

Membuat bangunan dari direktori file

Untuk membuat build game yang disimpan di lokasi mana pun, termasuk direktori lokal, gunakan [upload-build](#) AWS CLI perintah. Perintah ini membuat catatan build baru di Amazon GameLift Servers dan mengunggah file dari lokasi yang Anda tentukan.

Kirim permintaan unggahan. Di jendela baris perintah, masukkan upload-build perintah dan parameter berikut.

```
aws gamelift upload-build \  
  --name user-defined name of build \  
  --operating-system supported OS \  
  --server-sdk-version server SDK for Amazon GameLift Servers version \  
  --game-sdk-version game SDK for Amazon GameLift Servers version
```

```
--build-root build path \  
--build-version user-defined build number \  
--region region name
```

- **operating-system**— Lingkungan runtime build server game. Anda harus menentukan nilai OS. Anda tidak dapat memperbarui ini nanti.
- **server-sdk-version**— Versi dari Amazon GameLift Servers SDK server yang terintegrasi dengan server game Anda. Jika Anda tidak memberikan nilai, Amazon GameLift Servers menggunakan nilai default 4.0.2. Jika Anda menentukan versi SDK server yang salah, build server game mungkin gagal saat memanggil `InitSdk` untuk membuat sambungan ke Amazon GameLift Servers layanan.
- **build-root**— Jalur direktori file build Anda.
- **name**— Nama deskriptif untuk bangunan baru.
- **build-version**— Detail versi untuk file build.
- **region**— AWS Wilayah tempat Anda ingin membuat bangunan Anda. Buat build di Wilayah tempat Anda berencana untuk menyebarkan armada. Jika Anda menerapkan game di beberapa Wilayah, buat build di setiap Wilayah.

Note

Lihat Wilayah default Anda saat ini menggunakan [aws configure get region](#). Untuk mengubah Region default Anda, gunakan [aws configure set region *region name*](#) perintah.

Contoh

```
aws gamelift upload-build \  
  --operating-system AMAZON_LINUX_2023 \  
  
  --server-sdk-version "5.0.0" \  
  --build-root "~/mygame" \  
  --name "My Game Nightly Build" \  
  --build-version "build 255" \  
  --region us-west-2
```

```
aws gamelift upload-build \  
  --operating-system WINDOWS_2016 \  
  --server-sdk-version "5.0.0" \  
  --region us-east-1
```



```
--build-root "C:\mygame" \  
--name "My Game Nightly Build" \  
--build-version "build 255" \  
--region us-west-2
```

Menanggapi permintaan unggahan Anda, Amazon GameLift Servers menyediakan kemajuan unggahan. Pada unggahan yang berhasil, Amazon GameLift Servers mengembalikan ID catatan build baru. Waktu pengunggahan bergantung pada ukuran file game dan kecepatan koneksi Anda.

Membuat bangunan dengan file di Amazon S3

Anda dapat menyimpan file build Anda di Amazon S3 dan mengunggahnya Amazon GameLift Servers dari sana. Saat membuat build, Anda menentukan lokasi bucket S3, dan Amazon GameLift Servers mengambil file build langsung dari Amazon S3.

Untuk membuat sumber daya build

1. Simpan file build Anda di Amazon S3. Buat file.zip yang berisi file build yang dikemas dan unggah ke bucket S3 di file Anda. Akun AWS Catat label bucket dan nama file, Anda akan memerlukannya saat membuat Amazon GameLift Servers membangun.
2. Berikan Amazon GameLift Servers akses ke file build Anda. Buat peran IAM dengan mengikuti instruksi di [Mengakses file build game di Amazon S3](#). Setelah membuat peran, perhatikan Amazon Resource Name (ARN) peran baru, Anda akan memerlukannya saat membuat build.
3. Buat build. Gunakan Amazon GameLift Servers konsol atau AWS CLI untuk membuat catatan build baru. Anda harus memiliki PassRole izin, seperti yang dijelaskan dalam [Contoh izin IAM untuk Amazon GameLift Servers](#).

Console

1. Dalam [Amazon GameLift Servers konsol](#), di panel navigasi, pilih Hosting, Builds.
2. Pada halaman Builds, pilih Buat build.
3. Pada halaman Buat build, di bawah pengaturan Build, lakukan hal berikut:
 - a. Untuk Nama, masukkan nama skrip.
 - b. Untuk Versi, masukkan versi. Karena Anda dapat memperbarui konten build, data versi dapat membantu Anda melacak pembaruan.
 - c. Untuk Sistem operasi (OS), pilih OS build server game Anda. Anda tidak dapat memperbarui nilai ini nanti.

- d. Untuk build server Game, masukkan URI S3 dari objek build yang Anda upload ke Amazon S3, dan pilih versi Object. Jika Anda tidak ingat URI Amazon S3 dan versi objek, pilih Browse S3 dan cari objek build.
 - e. Untuk peran IAM, pilih peran yang Anda buat yang memberikan Amazon GameLift Servers akses ke bucket S3 Anda dan membangun objek.
4. (Opsional) Di bawah Tag, tambahkan tag ke build dengan memasukkan pasangan Kunci dan Nilai.
 5. Pilih Buat.

Amazon GameLift Servers menetapkan ID ke build baru dan mengunggah file.zip yang ditunjuk. Anda dapat melihat build baru, termasuk statusnya, di halaman Builds.

AWS CLI

Untuk menentukan build baru dan mengunggah file build server Anda, gunakan [create-build](#) perintah.

1. Buka jendela baris perintah dan beralih ke direktori tempat Anda dapat menggunakan file AWS CLI.
2. Masukkan create-build perintah berikut:

```
aws gamelift create-build \  
  --name user-defined name of build \  
  --server-sdk-version server SDK for Amazon GameLift Servers version \  
  --operating-system supported OS \  
  --build-version user-defined build number \  
  --storage-location "Bucket"=S3 bucket label, "Key"=Build .zip file  
name, "RoleArn"=Access role ARN} \  
  --region region name
```

- name— Nama deskriptif untuk bangunan baru.
- server-sdk-version— Versi SDK server untuk Amazon GameLift Servers Anda biasa mengintegrasikan server game Anda dengan Amazon GameLift Servers. Jika Anda tidak memberikan nilai, Amazon GameLift Servers menggunakan nilai default 4.0.2.
- operating-system— Lingkungan runtime build server game. Anda harus menentukan nilai OS. Anda tidak dapat memperbarui ini nanti.

- **build-version**— Detail versi untuk file build. Informasi ini dapat berguna karena setiap versi baru server game Anda memerlukan sumber daya build baru.
- **storage-location**
 - **Bucket**— Nama bucket S3 yang berisi build Anda. Misalnya, "my_build_files".
 - **Key**— Nama file.zip yang berisi file build Anda. Misalnya, "my_game_build_7.0.1, 7.0.2".
 - **RoleARN**— ARN ditugaskan ke peran IAM yang Anda buat. Misalnya, "arn:aws:iam: :111122223333:role/". GameLiftAccess Untuk contoh kebijakan, lihat [Mengakses file build game di Amazon S3](#).
- **region**— Buat build di AWS Wilayah tempat Anda berencana untuk menyebarkan armada. Jika Anda menerapkan game di beberapa Wilayah, buat build di setiap Wilayah.

 Note

Sebaiknya periksa Wilayah default Anda saat ini menggunakan [configure get](#) perintah . Untuk mengubah Region default Anda, gunakan [configure set](#) perintah.

Contoh

```
aws gamelift create-build \  
  --operating-system WINDOWS_2016 \  
  --storage-location  
  "Bucket"="my_game_build_files", "Key"="mygame_build_101.zip", "RoleArn"="arn:aws:iam::111122223333:role/  
gamelift" \  
  --name "My Game Nightly Build" \  
  --build-version "build 101" \  
  --region us-west-2
```

3. Untuk melihat build baru, gunakan [describe-build](#) perintah.

Perbarui build server game untuk Amazon GameLift Servers hosting terkelola

Saat Anda menerapkan build server game untuk Amazon GameLift Servers EC2 hosting terkelola, Anda mengunggah perangkat lunak server game Anda dan membuat Amazon GameLift Servers membangun sumber daya. Setelah Anda membuat Amazon GameLift Servers build, Anda dapat

memperbarui metadata build, tetapi Anda tidak dapat memperbarui file build itu sendiri. Untuk menyebarkan pembaruan ke server game Anda, unggah file yang diperbarui dan buat yang baru Amazon GameLift Servers membangun menggunakan AWS CLI perintah [upload-build](#) perintah. Atau, Anda dapat menggunakan [create-build](#) perintah untuk mengunggah build baru dari bucket Amazon S3 yang Anda kontrol. Kemudian gunakan build baru dengan membuat armada baru untuk itu.

Anda dapat memperbarui metadata build, termasuk nama dan deskripsi. Untuk tugas-tugas ini, gunakan Amazon GameLift Servers konsol atau [update-build](#) AWS CLI perintah.

Otomatiskan pembaruan pembuatan game Anda

Ikuti tips ini untuk membantu mengotomatiskan dan merampingkan proses memperbarui build server game Amazon GameLift Servers armada terkelola:

- Gunakan antrian sesi permainan dan tukar armada sesuai kebutuhan. Saat mengirim permintaan sesi game ke Amazon GameLift Servers, tentukan antrian sesi permainan alih-alih armada tertentu. Dengan antrian, Anda dapat menambahkan armada dengan build baru dan menghapus armada lama sesuai kebutuhan. Untuk informasi selengkapnya, lihat [Mengelola penempatan sesi permainan dengan Amazon GameLift Servers mengantre](#).
- Gunakan alias untuk mentransfer pemain ke build game baru. Saat mengirim permintaan sesi game ke Amazon GameLift Servers, tentukan alias armada alih-alih ID armada. Untuk informasi selengkapnya, lihat [Buat sebuah Amazon GameLift Servers alias](#).
- Siapkan untuk pengembangan berulang. Selama pengembangan game, jelajahi opsi untuk menyiapkan lingkungan pengujian yang dihosting yang mendukung pengembangan berulang yang cepat. Lihat [Siapkan untuk pengembangan berulang dengan Amazon GameLift Servers Dimanapun](#).

Cobalah sumber daya ini dari [Amazon GameLift Servers Toolkit](#) di Github:

Alat pembaruan build cepat (hanya untuk pengembangan)

Alat ini membantu Anda memodifikasi build server game yang sudah digunakan pada komputasi dalam EC2 armada terkelola, menghemat waktu Anda selama iterasi pengembangan cepat. Alat ini memiliki beberapa opsi; Anda dapat mengganti seluruh pembuatan game atau mengubah file tertentu, dan Anda dapat mengelola cara memulai ulang proses server game setelah pembaruan. Anda juga dapat menggunakannya untuk memperbarui semua komputasi dalam armada atau menargetkan komputasi individu.

Kunjungi Amazon GameLift Servers Repo toolkit di Github untuk mendapatkan [alat pembaruan build cepat](#) di Github dan mempelajari lebih lanjut tentang cara menggunakannya.

Skrip sampel penyebaran produksi

Skrip ini menggambarkan bagaimana Anda dapat mengotomatiskan proses memperbarui build server game yang digunakan pada armada terkelola EC2 dalam produksi. Untuk menggunakan skrip ini, Anda Amazon GameLift Servers solusi hosting harus menggunakan alias untuk armada IDs abstrak. Skrip sampel menangani langkah-langkah dasar berikut: mengunggah build yang diperbarui, membuat build baru dan menyebarkan ke armada baru, mengarahkan lalu lintas pemain dari armada yang ada ke armada baru, dan menghapus armada lama. Sesuaikan skrip sampel untuk memenuhi persyaratan penerapan spesifik Anda.

Kunjungi Amazon GameLift Servers Repo toolkit di Github untuk mendapatkan [skrip contoh penerapan produksi](#) di Github dan mempelajari lebih lanjut tentang cara menggunakannya.

Membangun gambar kontainer untuk Amazon GameLift Servers

Topik ini menjelaskan cara membuat gambar kontainer dengan perangkat lunak server game Anda untuk digunakan Amazon GameLift Servers. Gambar kontainer server game mencakup server game yang dapat dieksekusi dan dependensi apa pun yang perlu dijalankan. Gambar kontainer server game digunakan dengan solusi hosting kontainer Amazon GameLift Servers terkelola. Untuk detail tentang membangun solusi lengkap, lihat:

- [Peta jalan pengembangan untuk hosting dengan kontainer Amazon GameLift Servers terkelola](#)
- [Cara kerja kontainer Amazon GameLift Servers](#)

Selesaikan tugas-tugas berikut untuk menyiapkan image kontainer server game Anda untuk diterapkan ke armada Amazon GameLift Servers kontainer. Sebelum memulai tugas-tugas ini, selesaikan mengintegrasikan kode server game Anda dengan SDK Amazon GameLift Servers server.

Topik

- [Buat gambar kontainer server game](#)
- [Dorong gambar kontainer ke Amazon ECR](#)

Buat gambar kontainer server game

Ikuti petunjuk ini saat bekerja pada platform berbasis Linux atau menggunakan Windows Subsystem for Linux (WSL) dengan Docker diinstal.

Untuk membuat image kontainer server game

1. Siapkan direktori kerja dengan perangkat lunak server game Anda. Pada mesin lokal, buat direktori kerja untuk mengatur file untuk wadah server game Anda. Gambar kontainer Anda menggunakan struktur file ini saat menerapkan kontainer ke Amazon GameLift Servers sumber daya untuk hosting. Misalnya:

```
[~/]$ mkdir -p work/glc/gamebuild && cd work && find .  
.  
./glc  
./glc/gamebuild
```

Note

Jika Anda mencoba fitur ini dan belum memiliki build server game yang berfungsi, coba contoh server game kami [SimpleServer](#), yang tersedia di GitHub.

2. Buat Dockerfile baru menggunakan template yang disediakan.
3. Ikuti petunjuk di template Dockerfile untuk memperbaruinya untuk Anda gunakan sendiri.
 - Perbarui gambar dasar sesuai kebutuhan.
 - Tetapkan variabel lingkungan untuk build server game Anda.
4. Bangun gambar kontainer. Jalankan `docker build`, tentukan nama repositori Anda sendiri. Misalnya:

```
[~/work/glc]$ docker build -t <local repository name>:<optional tag> .
```

Anda dapat melihat repositori dan gambar IDs menggunakan `docker images` perintah, seperti yang diilustrasikan dalam contoh ini:

Template Dockerfile untuk gambar wadah server game

Template ini berisi instruksi minimum yang dibutuhkan wadah server game untuk dapat digunakan dalam Amazon GameLift Servers armada. Ubah konten sesuai kebutuhan untuk server game Anda.

```
# 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"]
```

Dorong gambar kontainer ke Amazon ECR

Setelah Anda membuat gambar kontainer untuk penyebaran Amazon GameLift Servers, simpan gambar di repositori publik atau pribadi di Amazon ECR. Repositori ini diberi nilai URI, yang Amazon GameLift Servers digunakan untuk mengambil snapshot gambar untuk penyebaran ke armada kontainer.

Note

[Jika Anda belum memiliki repositori pribadi Amazon ECR, buatlah satu.](#)

Untuk mendorong gambar kontainer Anda ke Amazon ECR

1. Dapatkan kredensial Amazon ECR Anda. Sebelum mendorong image kontainer ke Amazon ECR, pertama-tama dapatkan AWS kredensial Anda dalam bentuk sementara dan berikan ke Docker. Docker membutuhkan kredensial ini untuk masuk.

```
[~/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. Salin URI dari [repositori pribadi Amazon ECR](#) yang ingin Anda gunakan.
3. Terapkan tag Amazon ECR ke gambar kontainer Anda.

Example

```
[~/work/glc]$ docker tag <IMAGE ID from above> <Amazon ECR private repository URI>:<optional tag>
```

4. Dorong gambar kontainer Anda ke Amazon ECR

Example

```
[~/work/glc]$ docker image push <Amazon ECR private repository URI>
```

Menyebarkan skrip untuk Amazon GameLift ServersRealtime

Saat Anda siap untuk menerapkan Amazon GameLift Servers Realtime untuk game Anda, unggah file skrip Realtime server yang sudah selesai keAmazon GameLift Servers. Lakukan ini dengan membuat sumber daya Amazon GameLift Servers skrip dan menentukan lokasi file skrip Anda. Anda juga dapat memperbarui file skrip server yang sudah disebarkan dengan mengunggah file baru untuk sumber daya skrip yang ada.

Saat Anda membuat sumber daya skrip baruAmazon GameLift Servers, berikan ID skrip unik (misalnya,script-1111aaaa-22bb-33cc-44dd-5555eeee66ff) dan unggah salinan file skrip. Waktu unggah tergantung pada ukuran file skrip Anda dan kecepatan koneksi Anda.

Setelah Anda membuat sumber daya skripAmazon GameLift Servers, gunakan skrip dengan Amazon GameLift Servers Realtime armada baru. Amazon GameLift Serversmenginstal skrip server Anda ke setiap instance dalam armada, menempatkan file skrip di/local/game.

Untuk memecahkan masalah aktivasi armada yang terkait dengan skrip server, lihat. [Masalah Amazon GameLift Servers armada debug](#)

Buat paket file skrip

Skrip server Anda dapat menyertakan satu atau lebih file yang digabungkan menjadi satu file.zip untuk diunggah. File.zip harus berisi semua file yang perlu dijalankan skrip Anda.

Anda dapat menyimpan file skrip zip di direktori file lokal atau di bucket Amazon Simple Storage Service (Amazon S3).

Unggah file skrip dari direktori lokal

Jika Anda memiliki file skrip yang disimpan secara lokal, Anda dapat mengunggahnya Amazon GameLift Servers dari sana. Untuk membuat sumber daya skrip, gunakan Amazon GameLift Servers konsol atau [AWS Command Line Interface \(AWS CLI\)](#).

GameLift Peladen Amazon console

Untuk membuat sumber daya skrip

1. Buka [konsol Amazon GameLift Servers](#).
2. Di panel navigasi, pilih Hosting, Skrip.
3. Pada halaman Skrip, pilih Buat skrip.
4. Pada halaman Buat skrip, di bawah Pengaturan skrip, lakukan hal berikut:
 - a. Untuk Nama, masukkan nama skrip.
 - b. (Opsional) Untuk Versi, masukkan informasi versi. Karena Anda dapat memperbarui konten skrip, data versi dapat membantu dalam melacak pembaruan.
 - c. Untuk sumber Script, pilih Unggah file.zip.
 - d. Untuk file Script, pilih Pilih file, telusuri file.zip yang berisi skrip Anda, lalu pilih file itu.
5. (Opsional) Di bawah Tag, tambahkan tag ke skrip dengan memasukkan pasangan Kunci dan Nilai.
6. Pilih Buat.

Amazon GameLift Servers memberikan ID ke skrip baru dan mengunggah file.zip yang ditunjuk. Anda dapat melihat skrip baru, termasuk statusnya, di halaman Skrip.

AWS CLI

Gunakan perintah [create-script](#) AWS CLI untuk menentukan skrip baru dan unggah file skrip server Anda.

Untuk membuat sumber daya skrip

1. Tempatkan file.zip ke dalam direktori tempat Anda dapat menggunakan file. AWS CLI
2. Buka jendela baris perintah dan beralih ke direktori tempat Anda menempatkan file.zip.
3. Masukkan create-script perintah dan parameter berikut. Untuk --zip-file parameter, pastikan untuk menambahkan string fileb:// ke nama file.zip. Ini mengidentifikasi file sebagai biner sehingga Amazon GameLift Servers memproses konten terkompresi.

```
aws gamelift create-script \  
  --name user-defined name of script \  
  --script-version user-defined version info \  
  --zip-file fileb://name of zip file \  
  --region region name
```

Contoh

```
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
```

Menanggapi permintaan Anda, Amazon GameLift Servers mengembalikan objek skrip baru.

4. Untuk melihat skrip baru, hubungi [describe-script](#).

Unggah file skrip dari Amazon S3

Anda dapat menyimpan file skrip Anda di ember Amazon S3 dan mengunggahnya Amazon GameLift Servers dari sana. Saat membuat skrip, Anda menentukan lokasi bucket S3 dan Amazon GameLift Servers mengambil file skrip dari Amazon S3.

Untuk membuat sumber daya skrip

1. Simpan file skrip Anda dalam ember S3. Buat file.zip yang berisi file skrip server Anda dan unggah ke bucket S3 di tempat Akun AWS yang Anda kontrol. Perhatikan objek Uri—Anda memerlukan ini saat membuat skrip. Amazon GameLift Servers

Note

Amazon GameLift Servers tidak mendukung pengunggahan dari bucket S3 dengan nama yang berisi periode (.).

2. Berikan Amazon GameLift Servers akses ke file skrip Anda. Untuk membuat peran AWS Identity and Access Management (IAM) yang memungkinkan Amazon GameLift Servers untuk mengakses bucket S3 yang berisi skrip server Anda, ikuti instruksi di [Menyiapkan peran layanan IAM untuk Amazon GameLift Servers](#). Setelah Anda membuat peran baru, perhatikan namanya, yang Anda butuhkan saat membuat skrip.
3. Buat skrip. Gunakan Amazon GameLift Servers konsol atau AWS CLI untuk membuat catatan skrip baru. Untuk membuat permintaan ini, Anda harus memiliki PassRole izin IAM, seperti yang dijelaskan dalam [Contoh izin IAM untuk Amazon GameLift Servers](#).

GameLift Peladen Amazon console

1. Di [Amazon GameLift Servers konsol](#), di panel navigasi, pilih Hosting, Skrip.
2. Pada halaman Skrip, pilih Buat skrip.
3. Pada halaman Buat skrip, di bawah Pengaturan skrip, lakukan hal berikut:
 - a. Untuk Nama, masukkan nama skrip.
 - b. (Opsional) Untuk Versi, masukkan informasi versi. Karena Anda dapat memperbarui konten skrip, data versi dapat membantu dalam melacak pembaruan.
 - c. Untuk sumber Script, pilih Amazon S3 URI.
 - d. Masukkan URI S3 dari objek skrip yang Anda unggah ke Amazon S3, lalu pilih versi Object. Jika Anda tidak ingat URI Amazon S3 dan versi objek, pilih Browse S3, lalu cari objek skrip.
4. (Opsional) Di bawah Tag, tambahkan tag ke skrip dengan memasukkan pasangan Kunci dan Nilai.
5. Pilih Buat.

Amazon GameLift Servers memberikan ID ke skrip baru dan mengunggah file.zip yang ditunjuk. Anda dapat melihat skrip baru, termasuk statusnya, di halaman Skrip.

AWS CLI

Gunakan perintah [create-script](#) AWS CLI untuk menentukan skrip baru dan unggah file skrip server Anda.

1. Buka jendela baris perintah dan beralih ke direktori tempat Anda dapat menggunakan file AWS CLI.
2. Masukkan create-script perintah dan parameter berikut. --storage-locationParameter menentukan lokasi bucket Amazon S3 dari file skrip Anda.

```
aws gamelift create-script \
  --name [user-defined name of script] \
  --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
```

Contoh

```
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
```

Menanggapi permintaan Anda, Amazon GameLift Servers mengembalikan objek skrip baru.

3. Untuk melihat skrip baru, hubungi [describe-script](#).

Perbarui file Amazon GameLift ServersRealtime skrip

Anda dapat memperbarui metadata untuk sumber daya skrip menggunakan Amazon GameLift Servers konsol atau perintah. [update-script](#) AWS CLI

Anda juga dapat memperbarui konten skrip untuk sumber daya skrip. Amazon GameLift Servers menyebarkan konten skrip ke semua instance armada yang menggunakan sumber daya skrip yang diperbarui. Saat skrip yang diperbarui diterapkan, instance menggunakannya saat memulai sesi permainan baru. Sesi game yang sudah berjalan pada saat pembaruan tidak menggunakan skrip yang diperbarui.

Untuk memperbarui file skrip

- Untuk file skrip yang disimpan secara lokal, untuk mengunggah file skrip.zip yang diperbarui, gunakan Amazon GameLift Servers konsol atau perintah. `update-script`
- Untuk file skrip yang disimpan dalam bucket Amazon S3, unggah file skrip yang diperbarui ke bucket S3. Amazon GameLift Servers secara berkala memeriksa file skrip yang diperbarui dan mengambilnya langsung dari bucket S3.

Menyiapkan armada hosting dengan Amazon GameLift Servers

Di bagian ini, Anda akan menemukan informasi tentang merancang, membangun, dan memelihara Amazon GameLift Servers armada untuk meng-host server game Anda. Lihat [Amazon GameLift Serversopsi hosting](#) untuk mempelajari lebih lanjut tentang solusi hosting yang Amazon GameLift Servers ditawarkan, termasuk yang menggunakan armada terkelola, EC2 armada Anywhere yang dikelola sendiri untuk perangkat keras lokal Anda, dan solusi hybrid yang menggunakan keduanya.

Topik

- [Karakteristik armada](#)
- [Cara kerja pembuatan Amazon GameLift Servers armada](#)
- [Amazon GameLift Servers EC2 armada terkelola](#)
- [Amazon GameLift Serversarmada kontainer terkelola](#)
- [Amazon GameLift Servers Armada di mana saja](#)
- [Abstrak sebuah Amazon GameLift Servers penunjukan armada dengan alias](#)

Karakteristik armada

Amazon GameLift ServersArmada adalah kumpulan sumber daya komputasi yang menjalankan server game Anda dan menyelenggarakan sesi permainan untuk pemain. Armada dapat bervariasi dalam jenis sumber daya komputasi yang Anda gunakan dan bagaimana armada dikelola. Ukuran armada — jumlah sesi permainan dan pemain yang dapat didukungnya — tergantung pada jumlah sumber daya komputasi yang Anda berikan. Semua Amazon GameLift Servers armada memiliki karakteristik sebagai berikut:

- Proses server game yang berjalan di semua armada terintegrasi dengan SDK server untuk Amazon GameLift Servers dan berkomunikasi dengan Amazon GameLift Servers layanan dengan cara yang sama. Server game melaporkan ketersediaannya untuk menyelenggarakan sesi permainan dan pemain, menanggapi permintaan untuk memulai atau menghentikan sesi permainan, dan interaksi lainnya.
- Amazon GameLift Serversmenangani penempatan sesi permainan untuk semua armada dengan cara yang sama. Amazon GameLift Serversmelacak status server game armada dan memilih dari server game yang tersedia untuk menjadi tuan rumah sesi permainan baru. Proses ini digunakan

apakah permainan Anda menempatkan sesi permainan pada satu armada atau menggunakan [antrian sesi permainan](#) untuk menyeimbangkan hosting di beberapa armada. Dengan antrian, Anda juga dapat menyesuaikan keputusan penempatan untuk mempertimbangkan faktor-faktor seperti biaya sumber daya dan latensi.

- Semua armada mendukung penggunaan mak FlexMatch comblang bekerja sama dengan antrian penempatan sesi permainan. Amazon GameLift Servers Layanan menerima permintaan pertandingan pemain, membentuk pertandingan, dan meneruskannya ke antrian sesi permainan untuk menemukan server game yang tersedia.
- Amazon GameLift Servers mengumpulkan berbagai metrik armada. Ini termasuk metrik status untuk proses komputasi dan server, serta metrik penggunaan untuk sesi permainan dan aktivitas pemain. Lihat daftar lengkap metrik yang tersedia di [Monitor Amazon GameLift Servers dengan Amazon CloudWatch](#).

Cara kerja pembuatan Amazon GameLift Servers armada

Saat Anda meminta armada baru, Amazon GameLift Servers mulailah alur kerja untuk membuat sumber daya armada. Saat menyelesaikan setiap langkah alur kerja, Amazon GameLift Servers memperbarui status armada dan memancarkan serangkaian acara untuk mengkomunikasikan kemajuan menuju pembuatan armada.

Amazon GameLift Servers menggunakan dua jenis acara. Peristiwa transisi negara armada menandai saat status armada berubah. Acara pembuatan armada menyediakan penanda tambahan untuk membantu masalah pembuatan debugging. Anda dapat melacak semua peristiwa menggunakan Amazon GameLift Servers konsol atau dengan memanggil operasi Amazon GameLift Servers API [DescribeFleetEvents](#). Anda juga dapat melacak status armada dan lokasi menggunakan [DescribeFleetAttributes](#) atau [DescribeFleetLocationAttributes](#).

Amazon GameLift Servers EC2 armada terkelola

Amazon GameLift Servers EC2 armada terkelola menawarkan sumber daya berbasis cloud untuk hosting produksi dengan Amazon GameLift Servers. Dengan armada yang dikelola, Anda mendapatkan fleksibilitas, keamanan, dan keandalan AWS Cloud sumber daya yang lebih dioptimalkan untuk hosting game multipemain. Bagian Amazon GameLift Servers layanan menyediakan perkakas manajemen host yang kuat.

EC2 Armada terkelola adalah seperangkat komputasi virtual yang Amazon GameLift Servers memiliki dan beroperasi atas nama Anda dan berdasarkan pilihan konfigurasi Anda. Komputasi adalah instans

Amazon Elastic Compute Cloud EC2 (Amazon) yang secara fisik terletak di atau Local Wilayah AWS Zones. Saat membuat armada, Anda memilih jenis EC2 instans untuk komputasi berdasarkan daya komputasi, memori, penyimpanan, kemampuan jaringan, dan faktor lainnya.

Dengan EC2 armada terkelola, Anda mengunggah build server game Anda ke Amazon GameLift Servers. Saat Anda membuat armada, layanan akan menyebarkan build Anda ke komputasi armada dan meluncurkan proses server game. Setiap proses server game yang diluncurkan membuat koneksi ke Amazon GameLift Servers layanan dan laporan kesiapan untuk menyelenggarakan sesi permainan.

Selain pengerahan armada, Amazon GameLift Servers menangani tugas-tugas manajemen host berikut sehingga Anda tidak perlu:

- Melacak status semua komputasi dalam armada dan menggantikan komputasi basi atau tidak sehat.
- Menangani otentikasi untuk komunikasi antara proses server dan Amazon GameLift Servers layanan.
- Secara otomatis memulai dan menghentikan proses server game pada setiap komputasi, berdasarkan konfigurasi runtime Anda.
- Menawarkan alat auto-scaling untuk menyesuaikan kapasitas armada secara dinamis untuk memenuhi permintaan pemain.
- Melaporkan metrik kinerja untuk EC2 instance armada.

Lihat topik berikut tentang cara menyiapkan dan memelihara EC2 armada terkelola:

- [Peta jalan pengembangan untuk hosting dengan Amazon GameLift Servers terkelola EC2](#)
- [Buat EC2 armada Amazon GameLift Servers terkelola](#)
- [Menskalakan kapasitas hosting game dengan Amazon GameLift Servers](#)
- [Sesuaikan Amazon GameLift Servers EC2 armada yang dikelola](#)
- [Perbarui konfigurasi Amazon GameLift Servers armada](#)

Alur kerja pembuatan EC2 armada terkelola

Untuk armada yang dikelola, Amazon GameLift Servers mengatur sumber daya armada dan juga menyebarkan satu set sumber daya komputasi dengan perangkat lunak server game Anda diinstal

dan dijalankan. Ketika alur kerja pembuatan selesai dan berhasil, armada memiliki satu EC2 instance aktif di Wilayah asal armada dan masing-masing satu di lokasi terpencil armada. Semua contoh memiliki permainan yang siap untuk menjadi tuan rumah sesi permainan.

1. Amazon GameLift Servers menciptakan sumber daya armada di Wilayah asal armada dan menetapkan kapasitas yang diinginkan di setiap lokasi menjadi satu (1) instance. Status armada dan lokasi diatur ke Baru.
2. Amazon GameLift Servers mulai menulis acara ke log peristiwa armada.
3. Amazon GameLift Servers menetapkan status armada untuk Mengunduh dan mulai mempersiapkan perangkat lunak server game untuk penyebaran.
 - a. Mendapatkan server game yang diunggah membangun dan mengekstrak file terkompresi.
 - b. Menjalankan skrip instalasi, jika disediakan.
 - c. Menetapkan status armada ke Validasi dan mulai memverifikasi bahwa tidak ada kesalahan yang terjadi saat mengunduh dan menginstal file build.
4. Amazon GameLift Servers menetapkan status armada ke Building, mengonfigurasi perangkat keras armada, dan mengalokasikan satu EC2 instance untuk setiap instance armada.
5. Amazon GameLift Servers menetapkan status armada ke Activating. Meluncurkan proses server game pada setiap instance (berdasarkan instruksi runtime armada) dan menguji konektivitas antara build dan Amazon GameLift Servers layanan.
6. Ketika proses server game pada setiap instance membuat koneksi dan melaporkan kesiapan untuk menyelenggarakan sesi game, Amazon GameLift Servers menetapkan status armada dan lokasi ke Aktif. Pada titik ini, armada dianggap siap untuk menjadi tuan rumah sesi permainan.

Buat EC2 armada Amazon GameLift Servers terkelola

Topik ini menjelaskan cara membuat EC2 armada Amazon GameLift Servers terkelola. Armada terkelola menggunakan instans komputasi Amazon Elastic Compute Cloud EC2 (Amazon) yang dioptimalkan untuk hosting game multipemain. Anda dapat membuat armada terkelola yang menyebarkan komputasi ke global ke dan Local Wilayah AWS Zones yang didukung oleh. Amazon GameLift Servers

Saat Anda membuat EC2 armada terkelola baru, proses pembuatan armada segera dimulai. Armada terkelola melewati beberapa fase saat Amazon GameLift Servers mempersiapkan pembuatan server game, menyebarkan EC2 instance dengan build yang diinstal, dan meluncurkan server game di setiap instans. Anda dapat memantau status armada di konsol atau menggunakan uring AWS

Command Line Interface (AWS CLI). Sebuah armada siap untuk menjadi tuan rumah sesi permainan ketika statusnya mencapai ACTIVE. Untuk informasi selengkapnya tentang pembuatan armada terkelola, lihat topik berikut:

- [Cara kerja pembuatan Amazon GameLift Servers armada](#)
- [Masalah Amazon GameLift Servers armada debug](#)

Untuk membuat EC2 armada yang dikelola

Gunakan Amazon GameLift Servers konsol atau AWS Command Line Interface (AWS CLI) untuk membuat EC2 armada terkelola.

Console

Di [Amazon GameLift Servers konsol](#), gunakan panel navigasi untuk membuka halaman Armada. Pilih Buat armada untuk memulai alur kerja pembuatan armada.

Langkah 1 Pilih jenis komputasi

Pilih EC2 opsi Dikelola dan pilih Berikutnya.

Langkah 2 Tentukan detail armada

Pada langkah ini, tentukan beberapa pengaturan di seluruh armada.

 Untuk konfigurasi armada minimal:

- Berikan nama armada.
- Pilih tipe biner dan tentukan build atau skrip yang diunggah.
- Lewati bagian tentang detail dan tag tambahan.

1. Isi bagian Detail Armada:

- a. Masukkan Nama armada. Sebaiknya gunakan pola penamaan armada yang memudahkan identifikasi jenis armada saat melihat daftar armada.
- b. Berikan Deskripsi singkat tentang armada.
- c. Untuk tipe Biner, pilih Build untuk menunjukkan bahwa Anda sedang menerapkan build server game kustom, atau pilih atau Script jika Anda menerapkan Amazon

GameLift Servers Realtime ke armada ini. Pilih build atau skrip yang diunggah dari daftar dropdown.

2. (Opsional) Atur Detail tambahan sesuai kebutuhan.
 - a. Jika server game yang dapat dieksekusi perlu mengakses AWS sumber daya lain di akun Anda, tentukan peran Instans IAM dengan izin yang diperlukan. Untuk informasi selengkapnya, termasuk cara mengotorisasi aplikasi sisi server lainnya (seperti CloudWatch agen), lihat. [Berkomunikasi dengan AWS sumber daya lain dari armada Anda](#) Pengaturan ini tidak dapat diubah setelah Anda membuat armada.

Anda harus membuat peran sebelum Anda membuat armada yang menggunakannya. Selain itu, untuk membuat armada dengan peran instans, AWS pengguna Anda harus memiliki PassRole izin IAM (lihat [Contoh izin IAM untuk Amazon GameLift Servers](#)).

- b. Aktifkan opsi Hasilkan sertifikat TLS untuk mengatur otentikasi dan enkripsi untuk game Anda, klien Game menggunakan sertifikat ini untuk mengautentikasi server game saat menghubungkan dan mengenkripsi semua komunikasi klien/server. Untuk setiap instance dalam armada yang mendukung TLS, Amazon GameLift Servers juga membuat entri DNS baru dengan sertifikat. Pengaturan ini tidak dapat diubah setelah Anda membuat armada.
 - c. Jika Anda ingin menggabungkan data metrik untuk armada ini dan lainnya, tentukan nama grup Metrik. Gunakan nama grup metrik yang sama untuk semua armada yang ingin Anda gabungkan bersama. Lihat metrik untuk grup metrik untuk melihat data agregat.
3. (Opsional) Tambahkan Tag ke sumber daya armada. Setiap tag terdiri dari kunci dan nilai opsional, yang keduanya Anda tentukan. Tetapkan tag ke AWS sumber daya yang ingin Anda kategorikan dengan cara yang berguna, seperti berdasarkan tujuan, pemilik, atau lingkungan. Pilih Tambahkan tag baru untuk setiap tag yang ingin Anda tambahkan.
4. Pilih Berikutnya untuk melanjutkan alur kerja.

Langkah 3 Tentukan rincian contoh

Pada langkah ini, tentukan jenis sumber daya hosting yang akan digunakan dan di mana Anda ingin menerapkannya. Dengan memilih beberapa lokasi, Anda dapat menyebarkan server game Anda ke lokasi geografis yang lebih luas, yang menempatkan mereka lebih dekat

dengan pemain Anda dan meminimalkan latensi. Tidak semua jenis EC2 instance tersedia di semua lokasi.

 Untuk konfigurasi armada minimal:

- Jangan tambahkan lokasi terpencil.
- Atur jenis armada yang disetel ke “Sesuai Permintaan”. Armada spot membutuhkan pekerjaan pengaturan tambahan.
- Setel jenis instance ke “c5.large”. Jenis instance yang umum digunakan ini tersedia di semua Wilayah AWS.

1. Dalam penyebaran Instance, tentukan lokasi dan jenis armada.
 - a. Pilih satu atau beberapa Lokasi tambahan tempat Anda ingin menerapkan instance armada. Lokasi terpencil ini ditambahkan ke lokasi asal armada (yang telah dipilih sebelumnya), yang merupakan Wilayah AWS tempat Anda membuat armada ini. Anda dapat memilih lokasi jarak jauh dari semua Wilayah AWS dan Local Zones yang Amazon GameLift Servers mendukung.

Untuk mempelajari lebih lanjut tentang lokasi yang didukung, termasuk cara menggunakan lokasi Wilayah AWS yang tidak diaktifkan secara default, lihat [Amazon GameLift Servers lokasi layanan](#) untuk hosting terkelola. Juga tinjau Amazon GameLift Servers [kuota](#) lokasi per armada.

- b. Pilih untuk menggunakan instans On-Demand atau Spot untuk armada ini. Untuk informasi selengkapnya tentang jenis armada, lihat [Instans Sesuai Permintaan versus Instans Spot](#).
2. Pilih konfigurasi EC2 Instans Amazon yang memenuhi kebutuhan Anda dan tersedia di semua lokasi yang Anda pilih. Daftar ini difilter berdasarkan lokasi Anda saat ini dan pilihan jenis armada. Anda dapat memfilternya lebih lanjut dengan faktor lain seperti keluarga tipe instance dan arsitektur. Setelah membuat armada, Anda tidak dapat mengubah jenis instance.

Beberapa lokasi memiliki opsi tipe instance terbatas. Jika jenis instans pilihan Anda tidak tersedia untuk semua lokasi, pilih nilai ketersediaan lokasi untuk melihat detail selengkapnya. Untuk mengakomodasi semua lokasi, Anda mungkin perlu membuat armada terpisah dengan jenis instance yang berbeda.

Untuk informasi selengkapnya tentang memilih jenis instans, lihat [Tipe instans](#). Untuk mempelajari lebih lanjut tentang arsitektur Amazon EC2 Arm, lihat [Prosesor AWS Graviton dan jenis instans Amazon EC2](#). Untuk daftar lengkap jenis instance yang didukung oleh Amazon GameLift Servers, lihat referensi API untuk [EC2InstanceType\(CreateFleet\(\)\)](#).

 Note

Instans Graviton Arm memerlukan Amazon GameLift Servers server yang dibangun di OS Linux. Server SDK 5.1.1 atau yang lebih baru diperlukan untuk C++ dan C#. Server SDK 5.0 atau yang lebih baru diperlukan untuk Go. Instans ini tidak memberikan out-of-the-box dukungan untuk instalasi Mono di Amazon Linux 2023 (AL2023) atau Amazon Linux 2 (). AL2

3. Pilih Berikutnya untuk melanjutkan alur kerja.

Langkah 4 Konfigurasi runtime

Pada langkah ini, jelaskan bagaimana Anda ingin setiap instance dalam armada menjalankan perangkat lunak server game Anda. Tentukan item baris proses server terpisah untuk setiap executable untuk dijalankan pada sebuah instance, dan putuskan berapa banyak dari setiap proses server yang akan dijalankan secara bersamaan. Buka port pada setiap instance untuk memungkinkan pemain terhubung langsung ke server game. Anda dapat memperbarui pengaturan armada ini kapan saja.

 Untuk konfigurasi armada minimal:

- Tentukan item baris proses server tunggal untuk server game Anda yang dapat dieksekusi. Jika server game Anda membutuhkan proses lain untuk berjalan, buat definisi untuk masing-masing ini juga.
- Gunakan jumlah default proses bersamaan (1) untuk setiap item baris.
- Lewati pengaturan aktivasi sesi game.
- Tentukan nomor port tunggal.
- Lewati pengaturan sumber daya sesi game.

1. Buat konfigurasi Runtime untuk menginstruksikan Amazon GameLift Servers cara menjalankan proses server pada setiap instance dalam armada. Anda dapat mengubah konfigurasi runtime armada kapan saja setelah penerapan.
 - a. Masukkan jalur Peluncuran ke file yang dapat dieksekusi di build Anda. Pada instance Windows, executable server game dibangun ke jalur. `C:\game`
 Pada instance Linux, server game dibangun untuk `/local/game`. Contoh: `C:\game\MyGame\server.exe,/local/game/MyGame/server.exe`,
 atau `MyRealtimeLaunchScript.js`.
 - b. Masukkan parameter Peluncuran opsional untuk diteruskan ke game Anda yang dapat dieksekusi. Contoh: `+sv_port 33435 +start_lobby`.
 - c. Tentukan jumlah proses Bersamaan yang akan dijalankan pada setiap instance. Untuk server game yang dapat dieksekusi, setiap proses dapat meng-host satu sesi game, sehingga proses bersamaan menentukan jumlah sesi game yang dapat dihosting instans secara bersamaan.

Tinjau Amazon GameLift Servers [kuota](#) pada proses server per instance. Kuota ini berlaku untuk total proses bersamaan untuk semua konfigurasi. Jika Anda mengonfigurasi armada untuk melebihi mereka, armada tidak dapat diaktifkan.

2. Gunakan default aktivasi sesi Game atau sesuaikan untuk game Anda. Jika konfigurasi runtime memerlukan beberapa proses server game bersamaan per instance, pengaturan ini menentukan seberapa cepat sesi game baru dapat dimulai.
 - a. Atur aktivasi sesi permainan bersamaan Max untuk membatasi jumlah server game pada instance yang sedang mempersiapkan sesi permainan baru. Pengaturan ini berguna saat meluncurkan beberapa sesi permainan baru yang intensif sumber daya dan dapat memengaruhi kinerja sesi permainan berjalan lainnya.
 - b. Tetapkan batas waktu aktivasi baru untuk mencerminkan jumlah waktu maksimum yang harus diambil sesi permainan baru untuk menyelesaikan aktivasi dan melaporkan siap menjadi tuan rumah pemain. Amazon GameLift Servers mengakhiri aktivasi sesi permainan jika melebihi nilai ini.
3. Buka pengaturan EC2 port untuk memungkinkan lalu lintas masuk mengakses proses server pada armada. Pengaturan ini tidak diperlukan untuk membuat armada, tetapi Anda perlu mengaturnya sebelum pemain dapat terhubung ke sesi permainan di armada.

Untuk setiap pengaturan port, pilih Jenis protokol transfer data yang akan digunakan untuk komunikasi antara klien game dan server game Anda. Menyediakan rentang Port

(dalam formatnnnnn[-nnnnn]) dan rentang alamat IP menggunakan notasi CIDR (seperti **0.0.0.0/0** yang memungkinkan akses ke siapa saja).

Jika Anda perlu mengatur beberapa rentang non-berturut-turut, buat beberapa pengaturan port.

4. Tentukan pengaturan sumber daya sesi Game opsional. Anda dapat memperbarui pengaturan ini kapan saja setelah penerapan.
 - a. Aktifkan atau nonaktifkan kebijakan perlindungan penskalaan Game untuk semua instance di armada. Selama acara scale-down, tidak Amazon GameLift Servers akan menghentikan instance armada yang dilindungi jika mereka menyelenggarakan sesi permainan aktif.
 - b. Tetapkan batas pembuatan Sumber Daya maksimum jika Anda ingin membatasi jumlah sesi permainan yang dapat dibuat oleh satu pemain selama rentang waktu tertentu.
5. Pilih Berikutnya untuk melanjutkan alur kerja.

Langkah 5 Tinjau dan buat

Tinjau pengaturan Anda sebelum membuat armada. Meskipun beberapa pengaturan dapat diperbarui nanti (lihat[Perbarui konfigurasi Amazon GameLift Servers armada](#)), perubahan pada pengaturan berikut tidak diizinkan setelah armada dibuat:

- Jenis komputasi: Anda tidak dapat mengonversi EC2 armada terkelola ke armada Anywhere.
- Membangun atau skrip: Untuk menerapkan pembaruan ke build atau skrip server game Anda, Anda harus membuat armada baru.
- Opsi tambahan, termasuk peran instans dan pembuatan sertifikat TLS.
- Detail instans, termasuk jenis armada (Spot atau Sesuai Permintaan) dan tipe EC2 instans.

Saat Anda siap untuk menyebarkan armada baru, pilih Buat. Amazon GameLift Servers segera memulai proses aktivasi armada, menetapkan ID unik dan menempatkan armada dalam NEW status. Lacak kemajuan armada dari halaman Armada. Lihat halaman detail untuk armada dan buka tab Acara.

Anda dapat menyesuaikan kapasitas hosting armada setelah armada mencapai status AKTIF. Amazon GameLift Servers awalnya menyebarkan armada dengan satu instance di setiap lokasi armada. dan Anda menyesuaikan kapasitas dengan menambahkan instance ke setiap lokasi.

Untuk informasi selengkapnya, lihat [Menskalakan kapasitas hosting game dengan Amazon GameLift Servers](#).

AWS CLI

Gunakan `create-fleet` perintah untuk membuat armada tipe EC2 komputasi. Amazon GameLift Servers membuat sumber daya armada di default Anda saat ini Wilayah AWS (atau Anda dapat menambahkan tag `--region` untuk menentukan yang berbeda Wilayah AWS).

Buat armada terkelola minimal

Contoh permintaan berikut membuat armada baru dengan pengaturan minimal yang diperlukan untuk menyebarkan armada dengan menjalankan server game yang dapat dihubungkan oleh klien game. Armada baru memiliki karakteristik ini:

- Ini menentukan build server game, yang telah diunggah ke Amazon GameLift Servers dan dalam READY status.
- Is menggunakan c5.large On-Demand Instances dengan sistem operasi yang cocok dengan build game yang dipilih.
- Ini mengatur rumah Wilayah AWS armada us-west-2 dan menyebarkan instance ke Wilayah itu saja.
- Berdasarkan konfigurasi runtime, setiap komputasi dalam armada menjalankan satu proses server game, yang berarti bahwa setiap komputasi hanya dapat meng-host satu sesi game pada satu waktu. Batas waktu aktivasi sesi game diatur ke nilai default 300 detik, dan tidak ada batasan jumlah aktivasi bersamaan.
- Pemain dapat terhubung ke server game menggunakan pengaturan port tunggal 33435.
- Semua fitur lainnya dimatikan atau menggunakan pengaturan default.

```
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"
```

Buat armada terkelola yang dikonfigurasi sepenuhnya

Contoh permintaan berikut membuat armada produksi dengan pengaturan untuk semua fitur opsional. Armada baru memiliki karakteristik ini:

- Ini menentukan build server game, yang telah diunggah ke Amazon GameLift Servers dan dalam READY status.
- Ini menggunakan c5.large On-Demand Instances dengan sistem operasi yang cocok dengan build game yang dipilih.
- Ini mengatur rumah Wilayah AWS armada us-west-2 dan menyebarkan instance ke Wilayah asal dan satu lokasi terpencil. sa-east-1
- Berdasarkan konfigurasi runtime:
 - Setiap komputasi dalam armada menjalankan 10 proses server game dengan parameter peluncuran yang sama, yang berarti bahwa setiap komputasi dapat menghosting hingga 10 sesi game secara bersamaan.
 - Pada setiap komputasi, hanya dua sesi permainan yang dapat diaktifkan pada saat yang sama. Mengaktifkan sesi permainan harus siap untuk menjadi tuan rumah pemain dalam waktu 300 detik (5 menit) atau dihentikan.
- Pemain dapat terhubung ke server game menggunakan port dalam kisaran berikut 33435 to 33535.
- Ini menghasilkan sertifikat TLS untuk komunikasi terenkripsi antara klien game dan server.
- Semua sesi permainan di armada memiliki perlindungan sesi permainan diaktifkan.
- Pemain individu dibatasi untuk membuat tiga sesi permainan baru dalam periode 15 menit.
- Metrik untuk armada ini termasuk dalam grup metrik AMERfleets, yang (untuk contoh ini) mengumpulkan metrik untuk sekelompok armada di Amerika Utara, Tengah, dan Selatan.

```
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 \  
  --ec2-inbound-permissions "FromPort=33435,ToPort=33435,IpRange=0.0.0.0/0,Protocol=UDP"
```

```
--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"
```

Jika permintaan `create-fleet` berhasil, Amazon GameLift Servers mengembalikan satu set atribut armada yang menyertakan pengaturan konfigurasi yang Anda minta dan ID armada baru. Amazon GameLift Servers kemudian memulai proses aktivasi armada dan menetapkan status armada dan status lokasi ke `New`. Anda dapat melacak status armada dan melihat informasi armada lainnya menggunakan perintah CLI ini:

- [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](#)

Anda dapat mengubah kapasitas armada dan pengaturan konfigurasi lainnya sesuai kebutuhan menggunakan perintah berikut:

- [update-fleet-attributes](#)
- [update-fleet-capacity](#)
- [update-fleet-port-settings](#)
- [update-runtime-configuration](#)

- [create-fleet-locations](#)
- [delete-fleet-locations](#)

Sesuaikan Amazon GameLift Servers EC2 armada yang dikelola

Topik di bagian ini menguraikan beberapa penyesuaian yang dapat Anda buat saat membangun solusi hosting dengan Amazon GameLift Servers EC2 armada yang dikelola. Mereka memberikan panduan dan praktik terbaik untuk membuat dan mengonfigurasi armada untuk menjadi tuan rumah game Anda.

Keputusan yang diperlukan meliputi:

- Di mana Anda harus menyebarkan sumber daya hosting? Latensi adalah faktor utama dalam memilih lokasi armada Anda, tetapi biaya juga bervariasi menurut lokasi.
- Jenis EC2 instance apa yang paling mendukung game Anda? Pilih dari jenis instans yang tersedia di semua lokasi armada Anda untuk menggunakan kombinasi terbaik arsitektur komputasi, memori, penyimpanan, dan kapasitas jaringan.
- Berapa ukuran tipe instans yang Anda butuhkan? Pilih ukuran jenis instans berdasarkan kebutuhan sumber daya (memori dan CPU) perangkat lunak server game Anda dan faktor lainnya.
- Haruskah armada Anda menggunakan Instans Sesuai Permintaan atau Spot? Pertimbangkan apakah Anda dapat memanfaatkan harga Spot yang lebih rendah mengingat caranya Amazon GameLift Servers menjaga terhadap kemungkinan interupsi sesi permainan.
- Bagaimana Anda ingin perangkat lunak server game Anda berjalan pada setiap instance armada? Konfigurasi runtime memberi tahu Amazon GameLift Servers perangkat lunak server apa yang harus dijalankan dan bagaimana caranya.

Topik

- [Pilih sumber daya komputasi untuk armada terkelola](#)
- [Kelola caranya Amazon GameLift Servers meluncurkan server game](#)

Pilih sumber daya komputasi untuk armada terkelola

Untuk menyebarkan server game Anda dan menyelenggarakan sesi game di cloud, Amazon GameLift Servers menyediakan armada terkelola yang menggunakan [sumber daya Amazon Elastic Compute Cloud EC2 \(Amazon\) yang disebut instans](#). Gunakan topik berikut untuk membantu

memutuskan jenis EC2 instance apa yang ingin Anda gunakan untuk solusi hosting terkelola Anda dan cara mengonfigurasinya untuk menjalankan perangkat lunak server game Anda.

 Note

Jika Anda berencana untuk menggunakan sumber daya hosting yang Anda miliki, baik perangkat keras lokal atau hosting berbasis cloud lainnya, pertimbangkan opsi untuk hosting hybrid dengan Amazon GameLift Servers Di mana saja. Lihat [Menyiapkan armada hosting dengan Amazon GameLift Servers](#).

Topik

- [Lokasi armada](#)
- [Instans Sesuai Permintaan versus Instans Spot](#)
- [Sistem operasi](#)
- [Tipe instans](#)
- [Kuota layanan](#)

Lokasi armada

Pertimbangkan lokasi geografis tempat Anda berencana untuk men-deploy server game Anda. Ketersediaan jenis instans bervariasi menurut Wilayah AWS dan Zona Lokal.

Untuk armada multi-lokasi, ketersediaan instans dan kuota bergantung pada kombinasi Wilayah asal armada dan lokasi terpencil yang dipilih. Untuk informasi lebih lanjut tentang lokasi armada, lihat [Amazon GameLift Servers lokasi layanan](#).

Untuk Amazon GameLift Servers Di mana saja armada, Anda menentukan lokasi perangkat keras fisik Anda. Untuk informasi selengkapnya tentang lokasi kustom, lihat [Lokasi untuk Amazon GameLift Servers Di Mana Saja](#).

Instans Sesuai Permintaan versus Instans Spot

Instans EC2 On-Demand Amazon dan Instans Spot menawarkan perangkat keras dan kinerja yang sama, tetapi keduanya berbeda dalam ketersediaan dan biaya.

Instans Sesuai Permintaan

Anda dapat memperoleh Instans On-Demand saat Anda membutuhkannya, dan menyimpannya selama yang Anda inginkan. Instans On-Demand memiliki biaya tetap, artinya Anda membayar jumlah waktu yang Anda gunakan, dan tidak ada komitmen jangka panjang.

Instans Spot

Instans Spot dapat menawarkan alternatif hemat biaya untuk Instans On-Demand dengan memanfaatkan kapasitas komputasi yang tidak terpakai. AWS Harga Instans Spot berfluktuasi berdasarkan penawaran dan permintaan untuk setiap jenis instans di setiap lokasi. AWS dapat mengganggu Instans Spot kapan pun membutuhkan kapasitas kembali. Amazon GameLift Servers menggunakan antrian dan FleetIQ algoritma untuk menentukan AWS bahwa akan mengganggu Instance Spot, itu menempatkan instance dalam keadaan daur ulang. Kemudian, ketika tidak ada sesi permainan aktif pada instance, Amazon GameLift Servers mencoba untuk menggantikannya.

Untuk informasi selengkapnya tentang cara menggunakan Instans Spot, lihat [Desain antrian untuk Instans Spot](#).

Sistem operasi

Amazon GameLift Servers instance mendukung build server game yang berjalan di Microsoft Windows atau Amazon Linux. Saat Anda mengunggah build game ke Amazon GameLift Servers, tentukan sistem operasi untuk game. Saat Anda membuat EC2 armada Amazon untuk menyebarkan build game, Amazon GameLift Servers secara otomatis menyiapkan instance dengan sistem operasi build. Untuk informasi selengkapnya tentang sistem operasi server game yang didukung, lihat [Dapatkan alat Amazon GameLift Servers pengembangan](#).

Saat menggunakan Amazon GameLift Servers Di mana saja armada, Anda dapat menggunakan sistem operasi apa pun yang didukung perangkat keras Anda. Amazon GameLift Servers Armada di mana saja mengharuskan Anda untuk menyebarkan build game Anda ke perangkat keras saat menggunakan Amazon GameLift Servers untuk mengelola sumber daya Anda di satu tempat.

Tipe instans

Jenis instans EC2 armada Amazon menentukan jenis perangkat keras yang digunakan instans. Jenis instans yang berbeda menawarkan kombinasi daya komputasi, memori, penyimpanan, dan kemampuan jaringan yang berbeda.

Saat memilih dari jenis instans yang tersedia untuk game Anda, pertimbangkan:

- Arsitektur komputasi server game Anda: x64 atau Arm (AWS Graviton).

 Note

Contoh Graviton Arm membutuhkan Amazon GameLift Servers server dibangun di OS Linux. Server SDK 5.1.1 atau yang lebih baru diperlukan untuk C ++ dan C #. Server SDK 5.0 atau yang lebih baru diperlukan untuk Go. Instans ini tidak memberikan out-of-the-box dukungan untuk instalasi Mono di Amazon Linux 2023 (AL2023) atau Amazon Linux 2 (). AL2

- Persyaratan komputasi, memori, dan penyimpanan build server game Anda.
- Jumlah proses server yang Anda rencanakan untuk dijalankan per instance.

Dengan menggunakan jenis instans yang lebih besar, Anda mungkin dapat menjalankan beberapa proses server pada setiap instance. Ini dapat mengurangi jumlah contoh yang diperlukan untuk memenuhi permintaan pemain.

Untuk informasi selengkapnya:

- Tentang jenis instans, lihat [Jenis EC2 Instance Amazon](#).
- Tentang menjalankan beberapa proses per instance, lihat [Kelola caranya Amazon GameLift Servers meluncurkan server game](#).

Kuota layanan

Untuk melihat kuota layanan default untuk Amazon GameLift Servers, dan kuota saat ini untuk Anda Akun AWS, lakukan hal berikut:

- Untuk informasi kuota layanan umum untuk Amazon GameLift Servers, lihat [Amazon GameLift Servers titik akhir dan kuota](#) di. Referensi Umum AWS
- Untuk daftar jenis instans yang tersedia per lokasi untuk akun Anda, buka halaman [Kuota Layanan](#) Amazon GameLift Servers konsol. Halaman ini juga menampilkan penggunaan akun Anda saat ini untuk setiap jenis instans di setiap lokasi.
- Untuk daftar kuota akun Anda saat ini untuk jenis instans per Wilayah, jalankan perintah AWS Command Line Interface [describe-ec2-instance-limits](#)(AWS CLI). Perintah ini mengembalikan jumlah instance aktif yang Anda miliki di Wilayah default (atau di Wilayah lain yang Anda tentukan).

Saat Anda bersiap untuk meluncurkan game Anda, isi kuesioner peluncuran di [Amazon GameLift Servers konsol](#). Bagian Amazon GameLift Servers tim menggunakan kuesioner peluncuran untuk menentukan kuota dan batas yang benar untuk permainan Anda.

Kelola caranya Amazon GameLift Servers meluncurkan server game

Anda dapat menyiapkan konfigurasi runtime EC2 armada terkelola untuk menjalankan beberapa proses server game per instance. Ini menggunakan sumber daya hosting Anda dengan lebih efisien.

Bagaimana armada mengelola beberapa proses

Amazon GameLift Servers menggunakan konfigurasi runtime armada untuk menentukan jenis dan jumlah proses yang akan dijalankan pada setiap instance. Konfigurasi runtime berisi setidaknya satu konfigurasi proses server yang mewakili satu server game yang dapat dieksekusi. Anda dapat menentukan konfigurasi proses server tambahan untuk menjalankan jenis proses lain yang terkait dengan game Anda. Setiap konfigurasi proses server berisi informasi berikut:

- Nama file dan jalur yang dapat dieksekusi di bangunan game Anda.
- (Opsional) Parameter yang akan diteruskan ke proses saat peluncuran.
- Jumlah proses untuk berjalan secara konkuren.

Ketika sebuah instance dalam armada diaktifkan, ia meluncurkan serangkaian proses server yang ditentukan dalam konfigurasi runtime. Dengan berbagai proses, Amazon GameLift Servers terhuyung-huyung peluncuran setiap proses. Proses server memiliki rentang hidup yang terbatas. Saat mereka berakhir, Amazon GameLift Servers meluncurkan proses baru untuk mempertahankan jumlah dan jenis proses server yang ditentukan dalam konfigurasi runtime.

Anda dapat mengubah konfigurasi waktu aktif kapan saja dengan menambahkan, mengubah, atau menghapus konfigurasi proses server. Setiap instance secara teratur memeriksa pembaruan konfigurasi runtime armada untuk mengimplementasikan perubahan. Begini caranya Amazon GameLift Servers mengadopsi perubahan konfigurasi runtime:

1. Instance mengirimkan permintaan ke Amazon GameLift Servers untuk versi terbaru dari konfigurasi runtime.
2. Instance membandingkan proses aktifnya dengan konfigurasi runtime terbaru, dan kemudian melakukan hal berikut:
 - Jika konfigurasi runtime yang diperbarui menghapus jenis proses server, maka proses server aktif jenis ini terus berjalan hingga berakhir. Instance tidak menggantikan proses server ini.

- Jika konfigurasi runtime yang diperbarui mengurangi jumlah proses bersamaan untuk jenis proses server, maka kelebihan proses server jenis ini terus berjalan hingga berakhir. Instance tidak menggantikan proses server berlebih ini.
- Jika konfigurasi runtime yang diperbarui menambahkan jenis proses server baru atau meningkatkan proses bersamaan untuk tipe yang ada, maka instance memulai proses server baru, hingga Amazon GameLift Servers maksimum. Dalam hal ini, instance meluncurkan proses server baru saat proses yang ada berakhir.

Optimalkan armada untuk beberapa proses

Untuk menggunakan beberapa proses pada armada, lakukan hal berikut:

- [Buat build](#) yang berisi executable server game yang ingin Anda terapkan ke armada, lalu unggah build ke Amazon GameLift Servers. Semua server game dalam build harus berjalan pada platform yang sama dan menggunakan SDK server untuk Amazon GameLift Servers.
- Buat konfigurasi waktu aktif dengan satu atau lebih konfigurasi proses server dan beberapa proses konkuren.
- Integrasikan klien game dengan versi AWS SDK 2016-08-04 atau yang lebih baru.

Untuk mengoptimalkan kinerja armada, kami sarankan Anda melakukan hal berikut:

- Menangani skenario shutdown proses server sehingga Amazon GameLift Servers dapat mendaur ulang proses secara efisien. Sebagai contoh:
 - Tambahkan prosedur shutdown ke kode server game Anda yang memanggil API `ProcessEnding()` server.
 - Menerapkan fungsi callback `OnProcessTerminate()` dalam kode server game Anda untuk menangani permintaan penghentian Amazon GameLift Servers.
- Pastikan bahwa Amazon GameLift Servers menutup dan meluncurkan kembali proses server yang tidak sehat. Laporkan status kesehatan kembali ke Amazon GameLift Servers dengan menerapkan fungsi `OnHealthCheck()` callback dalam kode server game Anda. Amazon GameLift Servers secara otomatis mematikan proses server yang dilaporkan tidak sehat selama tiga laporan berturut-turut. Jika Anda tidak menerapkan `OnHealthCheck()`, maka Amazon GameLift Servers mengasumsikan bahwa proses server sehat, kecuali proses gagal merespons komunikasi.

Pilih jumlah proses per instance

Saat memutuskan jumlah proses bersamaan yang akan dijalankan pada sebuah instance, ingatlah hal berikut:

- Amazon GameLift Servers membatasi setiap instance ke [jumlah maksimum proses bersamaan](#). Jumlah semua proses bersamaan untuk konfigurasi proses server armada tidak dapat melebihi kuota ini.
- Untuk mempertahankan tingkat kinerja yang dapat diterima, jenis EC2 instans Amazon mungkin membatasi jumlah proses yang dapat berjalan secara bersamaan. Uji konfigurasi yang berbeda untuk game Anda untuk menemukan jumlah proses yang tepat untuk jenis instans pilihan Anda.
- Amazon GameLift Servers tidak menjalankan lebih banyak proses bersamaan daripada jumlah total yang dikonfigurasi. Ini berarti bahwa transisi dari konfigurasi runtime sebelumnya ke konfigurasi baru mungkin terjadi secara bertahap.

Perbarui konfigurasi Amazon GameLift Servers armada

Gunakan Amazon GameLift Servers konsol atau AWS CLI untuk memperbarui pengaturan armada Anda, mengubah lokasi jarak jauh, atau menghapus armada. Untuk armada terkelola, Anda tidak dapat mengubah build server game atau jenis instance armada. Sebagai gantinya, Anda harus mengganti armada.

Alat Pembaruan Bangun Cepat (hanya untuk pengembangan)

Dengan EC2 armada terkelola, untuk menerapkan pembaruan build server game, Anda perlu mengunggah setiap build baru Amazon GameLift Servers dan membuat armada baru untuknya.

Alat Pembaruan Bangun Cepat memungkinkan Anda melewati langkah-langkah ini selama pengembangan, menghemat waktu Anda dan memungkinkan iterasi pengembangan yang lebih cepat. Dengan alat ini, Anda dapat dengan cepat memperbarui file build game Anda di semua komputasi dalam armada yang ada. Alat ini memiliki beberapa opsi; Anda dapat mengganti seluruh game build atau mengubah 6 file tertentu, dan Anda dapat mengelola cara memulai ulang proses server game setelah pembaruan. Anda juga dapat menggunakannya untuk memperbarui komputasi individu dalam armada.

Untuk mendapatkan Fast Build Update Tool dan mempelajari lebih lanjut tentang cara menggunakannya, kunjungi repo Amazon GameLift Servers Toolkit untuk The [Fast Build Update Tool](#) di Github.

Anda dapat memperbarui atribut armada yang dapat berubah, pengaturan port, dan konfigurasi runtime menggunakan Amazon GameLift Servers konsol atau CLI. AWS Untuk mengubah batas penskalaan, lihat [Kapasitas armada skala otomatis dengan Amazon GameLift Servers](#).

GameLift Peladen Amazon console

1. Di [Amazon GameLift Serverskonsol](#), di panel navigasi, pilih Armada.
2. Pilih armada yang ingin Anda perbarui. Armada harus dalam ACTIVE status sebelum Anda dapat mengeditnya.
3. Pada halaman detail Armada, di salah satu bagian berikut, pilih Edit.
 - Pengaturan armada
 - Ubah atribut armada seperti Nama dan Deskripsi.
 - Menambahkan atau menghapus grup Metrik, yang CloudWatch digunakan Amazon untuk melacak Amazon GameLift Servers metrik agregat untuk beberapa armada.
 - Perbarui pengaturan batas pembuatan Sumber Daya.
 - Aktifkan atau nonaktifkan perlindungan sesi game.
 - Konfigurasi runtime - Anda dapat mengubah salah satu pengaturan konfigurasi runtime berikut dan menambah atau menghapus konfigurasi runtime.
 - Ubah jalur Peluncuran server game Anda.
 - Tambahkan, hapus, atau ubah parameter Peluncuran opsional.
 - Ubah jumlah proses Bersamaan yang dijalankan server game Anda.
 - Aktivasi sesi game — Ubah cara Anda ingin proses server berjalan dan host sesi game dengan memperbarui aktivasi sesi game bersamaan Max dan batas waktu aktivasi baru.
 - EC2 pengaturan port — Perbarui alamat IP dan rentang port yang memungkinkan akses masuk ke armada.
4. Pilih Konfirmasi untuk menyimpan perubahan.

AWS CLI

Gunakan perintah AWS CLI berikut untuk memperbarui armada:

- [update-fleet-attributes](#)
- [update-fleet-port-settings](#)
- [update-runtime-configuration](#)

Perbarui lokasi armada

Anda dapat menambah atau menghapus lokasi terpencil armada menggunakan Amazon GameLift Servers konsol atau AWS CLI. Anda tidak dapat mengubah wilayah asal armada.

GameLift Peladen Amazon console

1. Di [Amazon GameLift Servers konsol](#), di panel navigasi, pilih Armada.
2. Pilih armada yang ingin Anda perbarui. Armada harus dalam ACTIVE status sebelum Anda dapat mengeditnya.
3. Pada halaman detail Armada, pilih tab Lokasi untuk melihat lokasi armada.
4. Untuk menambahkan lokasi jarak jauh baru, pilih Tambah dan pilih lokasi yang ingin Anda gunakan instance. Daftar ini tidak menyertakan instance di mana jenis instance armada tidak tersedia.
5. Dengan lokasi baru yang dipilih, pilih Tambah. Amazon GameLift Servers menambahkan lokasi baru ke daftar, dengan status disetel ke NEW. Amazon GameLift Servers kemudian mulai menyediakan instance di setiap lokasi yang ditambahkan dan mempersiapkannya untuk menyelenggarakan sesi permainan.
6. Untuk menghapus lokasi terpencil yang ada dari armada, gunakan kotak centang untuk memilih satu atau beberapa lokasi yang terdaftar.
7. Dengan satu atau beberapa armada yang dipilih, pilih Hapus. Lokasi yang dihapus tetap ada dalam daftar, dengan status disetel ke DELETING. Amazon GameLift Servers kemudian memulai proses penghentian aktivitas di lokasi yang dihapus. Jika ada instance aktif yang menghosting sesi game, Amazon GameLift Servers gunakan proses penghentian server game untuk mengakhiri sesi game dengan anggun, menghentikan server game, dan mematikan instance.

AWS CLI

Gunakan perintah AWS CLI berikut untuk memperbarui lokasi armada:

- [create-fleet-locations](#)
- [delete-fleet-locations](#)

Hapus armada

Anda dapat menghapus armada saat Anda tidak lagi membutuhkannya. Menghapus armada secara permanen akan menghapus semua data yang terkait dengan sesi permainan dan sesi pemain, serta mengumpulkan data metrik. Sebagai alternatif, Anda dapat mempertahankan armada, menonaktifkan auto-scaling, dan menskalakan armada secara manual ke 0 instans.

Note

Jika armada memiliki koneksi peering VPC, pertama-tama minta otorisasi dengan menelepon. [CreateVpcPeeringAuthorization](#) Amazon GameLift Servers menghapus koneksi peering VPC selama penghapusan armada.

Anda dapat menggunakan Amazon GameLift Servers konsol atau alat AWS CLI untuk menghapus armada.

GameLift Peladen Amazon console

1. Di [Amazon GameLift Servers konsol](#), di panel navigasi, pilih Armada.
2. Pilih armada yang ingin Anda hapus. Anda hanya dapat menghapus armada di ACTIVE atau ERROR status.
3. Pilih Hapus.
4. Di kotak dialog Hapus armada, konfirmasi penghapusan dengan memasukkan. **delete**
5. Pilih Hapus.

AWS CLI

Gunakan perintah AWS CLI berikut untuk menghapus armada:

- [hapus-armada](#)

Masalah Amazon GameLift Servers armada debug

Topik ini memberikan panduan tentang cara menyelesaikan masalah dengan EC2 armada Amazon GameLift Servers terkelola Anda.

Masalah pembuatan armada

Saat Anda membuat EC2 armada terkelola, Amazon GameLift Servers layanan akan memulai alur kerja yang membuat armada, menyebarkan EC2 instance dengan build server game yang diinstal, dan memulai proses server game di setiap instance. Untuk penerangan mendetail, lihat [Cara kerja pembuatan Amazon GameLift Servers armada](#). Armada tidak dapat meng-host sesi game dan pemain hingga mencapai status Aktif.

Anda dapat men-debug masalah yang mencegah armada menjadi aktif dengan mengidentifikasi fase pembuatan armada tempat masalah terjadi dan meninjau peristiwa dan log pembuatan armada. Jika log tidak menawarkan informasi yang berguna, ada kemungkinan bahwa masalahnya adalah karena kesalahan layanan internal. Dalam situasi ini, cobalah untuk membuat armada lagi. Jika masalah berlanjut, coba unggah ulang build game untuk menyelesaikan kemungkinan kerusakan file). Anda juga dapat menghubungi Amazon GameLift Servers dukungan atau memposting pertanyaan di forum.

Mengunduh dan memvalidasi build

Selama fase ini, Amazon GameLift Servers buat server game yang Anda unggah, ekstrak file, dan jalankan skrip penginstalan apa pun. Jika pembuatan armada gagal selama fase ini, lihat peristiwa armada dan log untuk menentukan masalahnya. Kemungkinan penyebabnya meliputi:

- Amazon GameLift Server tidak bisa mendapatkan file build terkompresi (eventFLEET_BINARY_DOWNLOAD_FAILED). Pastikan lokasi penyimpanan build dapat diakses, bahwa Anda membuat armada yang Wilayah AWS sama dengan build, dan yang Amazon GameLift Servers memiliki izin yang benar untuk mengaksesnya.
- Amazon GameLift Server tidak dapat mengekstrak file build (eventFLEET_CREATION_EXTRACTING_BUILD).
- Skrip penginstalan dalam file build gagal diselesaikan dengan sukses (eventFLEET_CREATION_FAILED_INSTALLER).

Membangun sumber daya armada

Masalah selama fase ini biasanya melibatkan alokasi dan penyebaran sumber daya armada. Kemungkinan penyebabnya meliputi:

- Jenis instance yang diminta tidak tersedia.
- Jenis armada yang diminta (Spot atau Sesuai Permintaan) tidak tersedia.

Mengaktifkan proses server game

Selama fase ini, Amazon GameLift Servers sedang mencoba sejumlah tugas dan menguji elemen kunci, termasuk viabilitas server game, pengaturan konfigurasi runtime, dan kemampuan server game untuk terhubung dengan Amazon GameLift Servers layanan menggunakan Server SDK.

Note

Pada fase ini, Anda dapat mengakses instance armada dari jarak jauh untuk menyelidiki masalah lebih lanjut. Lihat [Terhubung dari jarak jauh ke Amazon GameLift Servers contoh armada](#).

Masalah yang mungkin terjadi meliputi:

- Proses server tidak mulai berjalan. Ini menunjukkan masalah dengan pengaturan konfigurasi runtime armada (peristiwa `FLEET_VALIDATION_LAUNCH_PATH_NOT_FOUND` atau `FLEET_VALIDATION_EXECUTABLE_RUNTIME_FAILURE`. Verifikasi bahwa Anda telah mengatur jalur peluncuran dan parameter peluncuran opsional dengan benar.
- Proses server mulai berjalan, tetapi armada gagal diaktifkan. Jika proses server mulai dan berjalan dengan sukses, tetapi armada tidak pindah ke status Aktif, kemungkinan penyebabnya adalah proses server gagal berkomunikasi dengan Amazon GameLift Servers layanan. Verifikasi bahwa server game Anda membuat panggilan SDK server yang benar ini (lihat [Inisialisasi proses server](#)):
 - Proses server gagal untuk menginisialisasi (`eventSERVER_PROCESS_SDK_INITIALIZATION_TIMEOUT`). Proses server tidak berhasil menelepon `InitSdk()`.
 - Proses server gagal memberi tahu Amazon GameLift Servers ketika siap untuk menyelenggarakan sesi permainan (`acaraSERVER_PROCESS_PROCESS_READY_TIMEOUT`). Proses server diinisialisasi tetapi tidak memanggil tepat `ProcessReady()` waktu.
- Permintaan koneksi peering VPC gagal. Untuk armada yang dibuat dengan koneksi peering VPC (lihat [Mengatur peering VPC dengan armada baru](#)), peering VPC dilakukan selama fase Mengaktifkan. Jika peering VPC gagal karena alasan apa pun, status armada baru tidak dapat berubah menjadi Aktif. Anda dapat melacak keberhasilan atau kegagalan permintaan peering dengan menelepon [describe-vpc-peering-connections](#). Pastikan untuk memeriksa apakah ada

otorisasi peering VPC yang valid ([describe-vpc-peering-authorizations](#)), karena otorisasi hanya berlaku selama 24 jam.

Masalah proses server

Proses server dimulai tetapi gagal dengan cepat atau melaporkan kondisi yang buruk.

Selain masalah dengan build game Anda, hasil ini dapat terjadi saat mencoba menjalankan terlalu banyak proses server secara bersamaan pada instans. Jumlah optimal proses serentak bergantung pada tipe instans dan persyaratan sumber daya server game Anda. Coba kurangi jumlah proses bersamaan, yang diatur dalam konfigurasi waktu aktif armada, untuk melihat apakah performa meningkat. Anda dapat mengubah konfigurasi runtime armada menggunakan Amazon GameLift Servers konsol (mengedit pengaturan alokasi kapasitas armada) atau dengan memanggil perintah. AWS CLI [update-runtime-configuration](#)

Masalah penghapusan armada

Armada tidak dapat diakhiri karena jumlah instans maksimum.

Pesan kesalahan menunjukkan bahwa armada yang dihapus masih memiliki instans aktif, yang tidak diizinkan. Anda harus terlebih dahulu menskalakan armada ke nol instans aktif. Ini dilakukan dengan secara manual menyetel jumlah instans yang diinginkan armada ke "0" dan kemudian menunggu penurunan skala dideploy. Pastikan untuk mematikan auto-scaling, yang akan melawan pengaturan manual.

Tindakan VPC tidak diotorisasi.

Masalah ini hanya berlaku untuk armada yang secara khusus Anda buat koneksi peering VPC (lihat. [VPC mengintip untuk Amazon GameLift Servers](#)). Skenario ini terjadi karena proses menghapus armada juga termasuk menghapus VPC armada dan koneksi peering VPC apa pun. Anda harus terlebih dahulu mendapatkan otorisasi dengan memanggil API layanan untuk Amazon GameLift Servers [CreateVpcPeeringAuthorization\(\)](#) atau menggunakan perintah AWS `create-vpc-peering-authorization` CLI. Setelah Anda memiliki otorisasi, Anda dapat menghapus armada.

Amazon GameLift ServersRealtimemasalah armada

Sesi game zombie: Mereka memulai dan menjalankan game, tetapi mereka tidak pernah berakhir.

Anda mungkin mengamati masalah ini sebagai salah satu skenario berikut:

- Pembaruan skrip tidak diambil oleh Realtime server armada.
- Armada dengan cepat mencapai kapasitas maksimum dan tidak menurunkan skala saat aktivitas pemain (seperti permintaan sesi game baru) berkurang.

Ini hampir pasti merupakan hasil dari kegagalan untuk berhasil memanggil `processEnding` Realtime skrip Anda. Meskipun armada berjalan aktif dan sesi game dimulai, tidak ada metode untuk menghentikannya. Akibatnya, Realtime server yang menjalankan sesi permainan tidak pernah dibebaskan untuk memulai yang baru, dan sesi permainan baru hanya dapat dimulai ketika Realtime server baru diputar. Selain itu, pembaruan Realtime skrip tidak memengaruhi sesi permainan yang sudah berjalan, hanya yang.

Untuk mencegah hal ini terjadi, skrip perlu menyediakan mekanisme untuk memicu panggilan `processEnding`. Seperti yang diilustrasikan dalam [Amazon GameLift ServersRealtimecontoh skrip](#), salah satu caranya adalah memprogram batas waktu sesi idle di mana, jika tidak ada pemain yang connect selama jangka waktu tertentu, skrip akan mengakhiri sesi game saat ini.

Namun, jika Anda termasuk dalam skenario ini, ada beberapa solusi untuk melepaskan server AndaRealtime. Triknya adalah dengan memicu proses Realtime server—atau instance armada yang mendasarinya—untuk memulai ulang. Dalam acara ini, Amazon GameLift Servers secara otomatis menutup sesi permainan untuk Anda. Setelah Realtime server dibebaskan, mereka dapat memulai sesi permainan baru menggunakan versi terbaru dari Realtime skrip.

Ada beberapa metode untuk mencapai ini, tergantung pada seberapa luas masalahnya:

- Menurunkan skala seluruh armada. Cara ini paling sederhana untuk dilakukan tetapi memiliki efek yang luas. Turunkan skala armada ke nol instans, tunggu hingga armada diturunkan sepenuhnya, lalu tingkatkan kembali. Ini akan menghapus semua sesi permainan yang ada, dan memungkinkan Anda memulai dengan Realtime skrip yang terbaru diperbarui.
- Akses instans dari jarak jauh dan mulai ulang prosesnya. Ini adalah pilihan yang baik jika Anda hanya memiliki beberapa proses untuk diperbaiki. Jika Anda sudah login ke instans, seperti tail log atau debug, maka ini mungkin menjadi metode tercepat. Lihat [Terhubung dari jarak jauh ke Amazon GameLift Servers contoh armada](#).

Jika Anda memilih untuk tidak menyertakan cara untuk memanggil `processEnding` Realtime skrip Anda, ada beberapa situasi rumit yang mungkin terjadi bahkan ketika armada aktif dan sesi permainan dimulai. Pertama, sesi game berjalan tidak berakhir. Akibatnya, proses server yang menjalankan sesi game tersebut tidak pernah bebas untuk memulai sesi game baru. Kedua, Realtime server tidak mengambil pembaruan skrip apa pun.

Terhubung dari jarak jauh ke Amazon GameLift Servers contoh armada

Anda dapat terhubung ke instans apa pun di aktif Anda Amazon GameLift Servers EC2 armada yang dikelola. Alasan umum untuk mengakses instance dari jarak jauh meliputi:

- Memecahkan masalah dengan integrasi server game Anda.
- Sempurnakan konfigurasi runtime Anda dan pengaturan khusus armada lainnya.
- Dapatkan aktivitas server game real-time, seperti pelacakan log.
- Jalankan alat benchmarking menggunakan lalu lintas pemain yang sebenarnya.
- Selidiki masalah tertentu dengan sesi permainan atau proses server.

Saat menghubungkan ke sebuah instance, pertimbangkan potensi masalah ini:

- Anda dapat terhubung ke instans apa pun dalam armada aktif. Umumnya, Anda tidak dapat terhubung ke armada non-aktif, seperti armada yang sedang dalam proses mengaktifkan atau berada dalam keadaan kesalahan. (Armada ini mungkin memiliki ketersediaan terbatas untuk waktu yang singkat.) Untuk bantuan terkait masalah aktivasi armada, lihat [Masalah Amazon GameLift Servers armada debug](#).
- Menghubungkan ke instans aktif tidak memengaruhi aktivitas hosting instans. Instance terus memulai dan menghentikan proses server berdasarkan konfigurasi runtime. Ini mengaktifkan dan menjalankan sesi permainan. Instance mungkin dimatikan sebagai respons terhadap peristiwa penurunan skala atau peristiwa lainnya.
- Setiap perubahan yang Anda buat pada file atau pengaturan pada instance dapat memengaruhi sesi permainan aktif instans dan pemain yang terhubung.

Instruksi berikut menjelaskan cara menghubungkan jarak jauh ke sebuah instance menggunakan antarmuka baris AWS perintah (CLI). Anda juga dapat melakukan panggilan terprogram menggunakan AWS SDK, seperti yang didokumentasikan dalam referensi API [layanan untuk Amazon GameLift Servers](#).

Kumpulkan data instance

Untuk terhubung ke Amazon GameLift Servers instance EC2 armada terkelola, Anda memerlukan informasi berikut:

- ID dari instance yang ingin Anda sambungkan. Anda dapat menggunakan ID instance atau ARN.
- SDK server untuk Amazon GameLift Servers versi yang digunakan pada instance. Server SDK terintegrasi dengan game build yang berjalan pada instance.

Petunjuk berikut menjelaskan bagaimana menyelesaikan tugas-tugas ini menggunakan AWS CLI. Anda harus mengetahui ID armada untuk contoh yang ingin Anda sambungkan.

1. Dapatkan nama komputasi. Dapatkan daftar semua komputasi aktif di armada. Panggil [daftar-komputasi dengan ID](#) armada atau ARN. Untuk armada satu lokasi, tentukan pengenal armada saja. Untuk armada multi-lokasi, tentukan pengenal armada dan lokasi. Dengan EC2 armada terkelola, `list-compute` mengembalikan daftar instance armada, dan properti `ComputeName` adalah ID instance. Temukan komputasi yang ingin Anda akses.

Permintaan

```
aws gamelift list-compute \
  --fleet-id "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa" \
  --location "sa-east-1"
```

Respons

```
{
  "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.uwp57060n1k6dnlnw49b78hg1
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. Temukan versi SDK server. Untuk informasi ini, Anda perlu mencari build yang dikerahkan ke armada. Versi SDK Server adalah properti build.
 - a. Hubungi [describe-fleet-attributes](#) dengan ID armada atau ARN untuk mendapatkan ID build armada dan ARN.
 - b. Panggil [describe-build](#) dengan ID build atau ARN untuk mendapatkan versi SDK server build.

Sebagai contoh:

Permintaan

```

aws gamelift describe-fleet-attributes /
--fleet-ids "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa"

```

Respons

```

{
  "FleetAttributes": [
    {
      "FleetId": "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
      "ComputeType": "EC2",
      "BuildId": "build-3333cccc-44dd-55ee-66ff-00001111aa22",
      . . .
    }
  ]
}

```

Permintaan

```

aws gamelift describe-build /
--build-id "build-3333cccc-44dd-55ee-66ff-00001111aa22"

```

Respons

```
"Build": {
  "BuildId": "build-1111aaaa-22bb-33cc-44dd-5555eeee66ff",
  "Name": "My_Game_Server_Build_One",
  "OperatingSystem": "AMAZON_LINUX_2023",

  "ServerSdkVersion": "5.1.1",
  . . .
}
```

Connect ke sebuah instans (server SDK 5)

Jika instance yang ingin Anda sambungkan menjalankan game build dengan server SDK versi 5.x, sambungkan ke instance menggunakan Amazon EC2 Systems Manager (SSM). Anda dapat mengakses instans jarak jauh yang menjalankan Windows atau Linux.

Sebelum Anda mulai:

Selesaikan langkah-langkah pengaturan SSM dan instal plugin SSM di mesin lokal Anda. Untuk informasi selengkapnya, lihat [Menyiapkan SSM](#) dan [Menginstal plugin Session Manager untuk AWS CLI](#) di Panduan Pengguna Amazon Systems EC2 Manager.

1. Minta kredensial akses untuk instance. Panggil [get-compute-access](#) dengan ID armada dan nama komputasi untuk instance yang ingin Anda sambungkan. Amazon GameLift Servers mengembalikan satu set kredensial sementara untuk mengakses instance. Sebagai contoh:

Permintaan

```
aws gamelift get-compute-access \
--compute-name i-11111111a222b333c \
--fleet-id fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa
--region us-west-2
```

Respons

```
{
  "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. Ekspor kredensial akses (opsional). Anda dapat mengekspor kredensial ke variabel lingkungan dan menggunakannya untuk mengkonfigurasi AWS CLI untuk pengguna default. Untuk detail selengkapnya, lihat [Variabel lingkungan untuk mengonfigurasi AWS CLI](#) di AWS Command Line Interface Panduan Pengguna.

```

export AWS_ACCESS_KEY_ID=ASIAIOSFODNN7EXAMPLE
export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY
export AWS_SESSION_TOKEN=AQoDYXdzEJr...<remainder of session token>

```

3. Connect ke instance armada. Mulai sesi SSM dengan instance yang ingin Anda sambungkan. Sertakan AWS Wilayah atau lokasi instance. Untuk informasi selengkapnya, termasuk cara mengatur SSM dan plugin SSM, lihat [Memulai sesi \(AWS CLI\) di Panduan Pengguna Amazon EC2 Systems Manager](#).

Permintaan sesi awal akan secara otomatis menggunakan kredensial yang Anda peroleh di Langkah 1.

```

aws ssm start-session \
--target i-11111111a222b333c \
--region us-west-2 \

```

Note

Jika Anda mendapatkan kesalahan akses ditolak, Anda mungkin memiliki variabel `AWS_PROFILE` lingkungan yang disetel ke AWS profil, yang menyebabkan AWS CLI menggunakan kredensial yang salah untuk akses jarak jauh. Untuk mengatasinya, hapus sementara variabel `AWS_PROFILE` lingkungan Anda. Atau, Anda dapat membuat AWS profil khusus untuk kredensial akses jarak jauh Anda dan menambahkan parameter baris `--profile` perintah ke permintaan `start-session`.

Connect ke sebuah instans (server SDK 4.x atau yang lebih lama)

Jika instance yang ingin Anda sambungkan menjalankan game build dengan server SDK versi 4 atau yang lebih lama, gunakan petunjuk berikut. Anda dapat terhubung ke instance yang menjalankan Windows atau Linux. Connect ke instance Windows menggunakan klien remote desktop protocol (RDP). Connect ke instance Linux menggunakan klien SSH.

1. Minta kredensial akses untuk instance. Bila Anda memiliki ID instance, gunakan perintah [get-instance-access](#) untuk meminta kredensial akses. Jika berhasil, Amazon GameLift Servers mengembalikan sistem operasi instance, alamat IP, dan satu set kredensial (nama pengguna dan kunci rahasia). Format kredensial tergantung pada sistem operasi instans. Gunakan instruksi berikut untuk mengambil kredensial, baik RDP maupun SSH.

- Untuk instans Windows – Untuk connect ke instans Windows, RDP memerlukan nama pengguna dan kata sandi. Permintaan `get-instance-access` mengembalikan nilai-nilai ini sebagai string sederhana, sehingga Anda dapat menggunakan nilai yang dikembalikan apa adanya. Kredensi contoh:

```
"Credentials": {
  "Secret": "aA1bBB2cCCd3EEE",
  "UserName": "gl-user-remote"
}
```

- Untuk instans Linux – Untuk connect ke instans Linux, SSH membutuhkan nama pengguna dan kunci privat. Amazon GameLift Servers mengeluarkan kunci pribadi RSA dan mengembalikannya sebagai string tunggal, dengan karakter baris baru (\n) menunjukkan jeda baris. Untuk membuat kunci pribadi dapat digunakan, ambil langkah-langkah ini: (1) konversi string ke .pem file, dan (2) mengatur izin untuk file baru. Contoh kredensial yang dikembalikan:

```
"Credentials": {
  "Secret": "-----BEGIN RSA PRIVATE KEY-----
nEXAMPLEKEYKCAQEAy7WZhaDs1A1W3mR1QtvhwyORRX8gnxgDAfRt/gx42kWXsT4rXE/b5CpSgie/
\nvBoU7jLxx92pNHoFnByP+Dc21eyyz6CvjTmWA0JwfWiW5/akH7i05dSrvC7dQkW2duV5QuUdEQW
\nZ/aNxMniGQE6XAgfwlnXVBwrerrQo+ZWQeqiUwwMkuEbLeJFLhMCvYURpUMSC1oehm449ilx9X1F
\nG50TCFe0zfl8dqqCP6GzbPaIjiU19xX/az0R9V+tpU0zEL+wmXnZt3/nHPQ5xvD20JH67km6SuPW
\noPzev/D8V+x4+bHthfSjR9Y7DvQFjFBVwHXigBdtZcU2/wei8D/HYwIDAQABAOIBAGZ1kaEvnrqu
\n/uler7vgIn5m71N5LKw4hJLAIW6tUT/fzvtcHK0SbkQCQXuriHmQ2MQyJX/0kn2NfjLV/
ufGxbL1\nmb5qwmGUnEpJaZD6QSSs3kICLwUYUiGfc0uisbmJoap/
GTLU0W5Mfcv36PaBUNy5p53V6G7hXb2\nbahyWyJNfjLe4M86yd2YK3V2CmK+X/
```

```

B0sShnJ36+hjrXPPWmV3N9zEmCdJjA+K15DYmhm/
tJWSD9\n81oGk9TopEp7CkIfatEATyyZiVqoRq6k64iuM9JkA30zdXzMQexXVJ1TLZVEH0E7bh1Y9d801ozR
\noQs/FiZNAx2iijCWyv01pjE73+kCgYEA9mZtyhkHkFDpwrSM1APaL8oNAbbjwEy7Z5Mqfq1
+1Ip1\nYkriL0DbLXlvRAH+yHPRit2hH0jtUNZh4Axv+cpq09qbUI3+43eEy24B7G/Uh
+GTfbjsXs0xQx/x\np9otyVwc7hsQ5TA5PZb
+mvkJ50BEKzet9XCkKw0NBYELGhnEPe7cCgYEA06Vgov6YHleHui9kHuws
\nayav0elc5zkxjF9nfHFJRry21R1trw2Vdpn+9g481URipzWV0Eihvm+xTtmaZ1Sp//1kq75XDwnU
\nWA8gkn603QE3fq2yN98BURsAKdJfJ5RL1HvGQvTe10HLYYXpJnEkHv+Unl2ajLivWUt5pbBrKbUC
\nngYBjb0+0Zk0sCcpZ29sbzjYjpIddErySIyRX5gV2uNQwAjLdp9PfN295yQ+BxMBXiIycWVQiw0bH
\nnoMo7yykABY70zd5wQewBQ4AdSlWSX4nGDtsiFxiWiI5sKuAAe0CbTosy1s8w8fxoJ5Tz1sdoxNeGs
\nArq6Wv/G16zQuAE9zK9vwwKBgF+09VI/1wJBirsDGz9whVwFFPrTkJNvJZzYt69qezx1sjgFKshy
\nWBhd4xHZtmCqpBP1AymEjr/T0lbxyARMXMnIOWIANXMG4KGSyl1mzSVAoQ+fqR+cJ3d0dyP11j
\njjb0Ed/NY8fr1NDxAVHE8BSkdsx2f6ELEyBKJSRr9snRAoGAMrTwYneXzvTskF/S5Fyu0i0egLda
\nNWU38v/nDCgEpIXD5Hn3qAEcju1IjmbwlvTW+nY2jVhv7UGd8MjwUTNGItdb6nsYqM2asrnF3qS
\nVRkAKKKYeGjKpUfVTTrW0YFjXkfcrR/V+QFL50ndHAKJXjW7a4ejJLncTzmZSpYzwApc=\n-----END
RSA PRIVATE KEY-----",
    "UserName": "gl-user-remote"
}

```

Saat menggunakan AWS CLI, Anda dapat secara otomatis membuat .pem file dengan menyertakan parameter `--query` dan `--output` ke permintaan Anda. `get-instance-access`

Gunakan perintah berikut untuk memeriksa izin pada .pem file.

```
$ chmod 400 MyPrivateKey.pem
```

2. Buka port untuk koneksi jarak jauh. Anda dapat mengakses instans di Amazon GameLift Servers armada melalui pelabuhan apa pun yang diizinkan dalam konfigurasi armada. Anda dapat melihat pengaturan port armada menggunakan perintah [describe-fleet-port-settings](#).

Sebagai praktik terbaik, kami menyarankan untuk membuka port untuk akses jarak jauh hanya saat Anda membutuhkannya dan menutupnya saat Anda selesai. Anda tidak dapat memperbarui pengaturan port setelah membuat armada tetapi sebelum aktif. Jika Anda macet, buat ulang armada dengan pengaturan port terbuka.

Gunakan perintah [update-fleet-port-settings](#) untuk menambahkan pengaturan port untuk koneksi jarak jauh (seperti 22 untuk SSH atau 3389 untuk RDP). Untuk nilai rentang IP, tentukan alamat IP untuk perangkat yang akan Anda gunakan untuk connect (dikonversi ke format CIDR). Contoh:

```
$ 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"
```

Contoh berikut membuka port 3389 pada armada 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. Buka klien koneksi jarak jauh. Gunakan Desktop Jarak Jauh untuk Windows atau SSH untuk instans Linux. Connect ke instans menggunakan alamat IP, pengaturan port, dan kredensial akses.

Contoh SSH:

```
ssh -i MyPrivateKey.pem gl-user-remote@192.0.2.0
```

Melihat file pada instance jarak jauh

Saat terkoneksi ke instans dari jarak jauh, Anda memiliki akses pengguna dan administratif penuh. Ini berarti Anda juga memiliki kemampuan untuk menyebabkan kesalahan dan kegagalan dalam hosting game. Jika instans menghosting game dengan pemain aktif, Anda berisiko menabrak sesi game dan menjatuhkan pemain, atau mengganggu proses shutdown game dan menyebabkan kesalahan dalam data dan log game yang disimpan.

Cari sumber daya ini pada instance hosting:

- Game membangun file. File-file ini adalah game build yang Anda unggah Amazon GameLift Servers. Mereka termasuk satu atau lebih server game executable, aset, dan dependensi. File build game ada di direktori root yang disebut game:
 - Pada Windows: c:\game
 - Pada Linux: /local/game
- File log game. Temukan file log yang dihasilkan server game Anda di direktori game root di jalur direktori apa pun yang Anda tentukan.

- Amazon GameLift Servers sumber daya hosting. Direktori root `Whitewater` berisi file yang digunakan oleh Amazon GameLift Servers layanan untuk mengelola aktivitas hosting game. Jangan memodifikasi file-file ini karena alasan apa pun.
- Konfigurasi runtime. Jangan mengakses konfigurasi runtime untuk instance individual. Untuk membuat perubahan pada properti konfigurasi runtime, perbarui konfigurasi runtime armada (lihat operasi AWS SDK [UpdateRuntimeConfiguration](#) atau). AWS CLI [update-runtime-configuration](#)
- Data armada. File JSON berisi informasi tentang armada yang dimiliki instance, untuk digunakan oleh proses server yang berjalan pada instance. File JSON berada di lokasi berikut:
 - Pada Windows: `C:\GameMetadata\gamelift-metadata.json`
 - Pada Linux: `/local/gamemetadata/gamelift-metadata.json`
- Sertifikat TLS. Jika instance berada di armada yang mengaktifkan pembuatan sertifikat TLS, cari file sertifikat, termasuk sertifikat, rantai sertifikat, kunci pribadi, dan sertifikat root di lokasi berikut:
 - Pada Windows: `c:\GameMetadata\Certificates`
 - Pada Linux: `/local/gamemetadata/certificates/`

VPC mengintip untuk Amazon GameLift Servers

Topik ini memberikan panduan tentang cara mengatur koneksi peering VPC antara Anda Amazon GameLift Servers-server game yang di-host dan non-Amazon GameLift Servers sumber daya. Gunakan koneksi peering Amazon Virtual Private Cloud (VPC) untuk memungkinkan server game Anda berkomunikasi secara langsung dan pribadi dengan AWS sumber daya Anda yang lain, seperti layanan web atau repositori. Anda dapat membuat peering VPC dengan sumber daya apa pun yang berjalan AWS dan dikelola oleh AWS akun yang dapat Anda akses.

Note

Peering VPC adalah fitur lanjutan. Untuk mempelajari tentang opsi pilihan untuk memungkinkan server game Anda berkomunikasi secara langsung dan pribadi dengan AWS sumber daya Anda yang lain, lihat. [Berkomunikasi dengan AWS sumber daya lain dari armada Anda](#)

Jika Anda sudah terbiasa dengan peering Amazon VPCs dan VPC, pahami bahwa mengatur peering dengan Amazon GameLift Servers server game agak berbeda. Anda tidak memiliki akses ke VPC yang berisi server game Anda—itu dikendalikan oleh Amazon GameLift Servers layanan—jadi Anda

tidak dapat langsung meminta VPC mengintip untuk itu. Sebagai gantinya, Anda terlebih dahulu melakukan pra-otorisasi VPC dengan non-Amazon GameLift Servers sumber daya untuk menerima permintaan peering dari Amazon GameLift Servers layanan. Kemudian Anda memicu Amazon GameLift Servers untuk meminta pengintip VPC yang baru saja Anda otorisasi. Amazon GameLift Servers menangani tugas-tugas membuat koneksi peering, mengatur tabel rute, dan mengkonfigurasi koneksi.

Menyiapkan peering VPC untuk armada yang sudah ada

1. Dapatkan ID AWS akun dan kredensial.

Anda memerlukan ID dan kredensi masuk untuk akun berikut. AWS Anda dapat menemukan AWS akun IDs dengan masuk ke [AWS Management Console](#) dan melihat pengaturan akun Anda. Untuk mendapatkan kredensial, buka konsol IAM.

- AWS akun yang Anda gunakan untuk mengelola Amazon GameLift Servers server game.
- AWS akun yang Anda gunakan untuk mengelola non-Amazon GameLift Servers sumber daya.

Jika Anda menggunakan akun yang sama untuk Amazon GameLift Servers dan non-Amazon GameLift Servers sumber daya, Anda memerlukan ID dan kredensial untuk akun itu saja.

2. Dapatkan pengenalan untuk setiap VPC.

Dapatkan informasi berikut untuk keduanya yang akan VPCs diintip:

- VPC untuk Anda Amazon GameLift Servers server game — Ini adalah milik Anda Amazon GameLift Servers ID armada. Server game Anda digunakan di Amazon GameLift Servers pada armada EC2 contoh. Armada secara otomatis ditempatkan di VPC sendiri, yang dikelola oleh Amazon GameLift Servers layanan. Anda tidak memiliki akses langsung ke VPC, sehingga diidentifikasi oleh ID armada.
- VPC untuk non-Amazon GameLift Servers AWS sumber daya — Anda dapat membuat peering VPC dengan sumber daya apa pun yang berjalan AWS dan dikelola oleh AWS akun yang dapat Anda akses. Jika Anda belum membuat VPC untuk sumber daya ini, lihat [Memulai Amazon VPC](#). Setelah Anda membuat VPC, Anda dapat menemukan ID VPC dengan masuk ke VPC for [AWS Management Console](#) Amazon dan melihat VPC Anda. VPCs

 Note

Saat mengatur peering, keduanya VPCs harus ada di wilayah yang sama. VPC untuk Anda Amazon GameLift Servers server game armada berada di wilayah yang sama dengan armada.

3. Mengotorisasi peering VPC.

Pada langkah ini, Anda melakukan pra-otorisasi permintaan future dari Amazon GameLift Servers untuk mengintip VPC dengan server game Anda dengan VPC Anda untuk non-Amazon GameLift Servers sumber daya. Tindakan ini memperbarui grup keamanan untuk VPC Anda.

Untuk mengotorisasi peering VPC, panggil layanan [CreateVpcPeeringAuthorizationAPI \(\)](#) atau gunakan perintah CLI. `AWS create-vpc-peering-authorization` Lakukan panggilan ini menggunakan akun yang mengelola non-Amazon GameLift Servers sumber daya. Tentukan informasi berikut:

- ID VPC Peer - Ini untuk VPC dengan non-Amazon GameLift Servers sumber daya.
- Amazon GameLift Servers AWS ID akun — Ini adalah akun yang Anda gunakan untuk mengelola Amazon GameLift Servers armada.

Setelah Anda mengesahkan peering VPC, otorisasi tetap berlaku selama 24 jam kecuali dicabut. Anda dapat mengelola otorisasi peering VPC menggunakan operasi berikut:

- [DescribeVpcPeeringAuthorizations\(\)](#) (AWS CLI `describe-vpc-peering-authorizations`).
- [DeleteVpcPeeringAuthorization\(\)](#) (AWS CLI `delete-vpc-peering-authorization`).

4. Meminta koneksi peering.

Dengan otorisasi yang valid, Anda dapat memintanya Amazon GameLift Servers membangun koneksi mengintip.

Untuk meminta peering VPC, panggil layanan API [CreateVpcPeeringConnection\(\)](#) atau gunakan perintah CLI AWS . `create-vpc-peering-connection` Lakukan panggilan ini menggunakan akun yang mengelola Amazon GameLift Servers server game. Gunakan informasi berikut untuk mengidentifikasi dua VPCs yang ingin Anda peer:

- ID VPC rekan AWS dan ID akun - Ini adalah VPC untuk non-Amazon GameLift Servers sumber daya dan akun yang Anda gunakan untuk mengelolanya. ID VPC harus sesuai dengan ID pada otorisasi peering yang valid.
- Fleet ID — Ini mengidentifikasi VPC untuk Anda Amazon GameLift Servers server game.

5. Lacak status koneksi peering.

Meminta koneksi peering VPC adalah operasi asynchronous. Untuk melacak status permintaan peering dan menangani kasus sukses atau gagal, gunakan salah satu opsi berikut:

- Polling terus menerus dengan `DescribeVpcPeeringConnections()`. Operasi ini mengambil catatan koneksi peering VPC, termasuk status permintaan. Jika koneksi peering berhasil dibuat, catatan koneksi juga berisi blok CIDR dari alamat IP privat yang ditetapkan ke VPC.
- Menangani peristiwa armada yang terkait dengan koneksi peering VPC dengan [DescribeFleetEvents\(\)](#), termasuk peristiwa keberhasilan dan kegagalan.

Setelah koneksi peering dibuat, Anda dapat mengelolanya menggunakan operasi berikut:

- [DescribeVpcPeeringConnections\(\)](#) (AWS CLI `describe-vpc-peering-connections`).
- [DeleteVpcPeeringConnection\(\)](#) (AWS CLI `delete-vpc-peering-connection`).

Mengatur peering VPC dengan armada baru

Anda dapat membuat yang baru Amazon GameLift Servers armada dan meminta koneksi peering VPC pada saat yang bersamaan.

1. Dapatkan ID AWS akun dan kredensial.

Anda memerlukan ID dan kredensial masuk untuk dua akun berikut. AWS Anda dapat menemukan AWS akun IDs dengan masuk ke [AWS Management Console](#) dan melihat pengaturan akun Anda. Untuk mendapatkan kredensial, buka konsol IAM.

- AWS akun yang Anda gunakan untuk mengelola Amazon GameLift Servers server game.
- AWS akun yang Anda gunakan untuk mengelola non-Amazon GameLift Servers sumber daya.

Jika Anda menggunakan akun yang sama untuk Amazon GameLift Servers dan non-Amazon GameLift Servers sumber daya, Anda memerlukan ID dan kredensial untuk akun itu saja.

2. Dapatkan ID VPC untuk non-Amazon GameLift ServersAWS sumber daya.

Jika Anda belum membuat VPC untuk sumber daya ini, lakukan sekarang (lihat [Memulai dengan Amazon VPC](#)). Pastikan bahwa Anda membuat VPC baru di wilayah yang sama dengan tempat Anda berencana untuk membuat armada baru Anda. Jika Anda tidak-Amazon GameLift Servers sumber daya dikelola di bawah AWS akun atau grup pengguna/pengguna yang berbeda dari yang Anda gunakan Amazon GameLift Servers, Anda harus menggunakan kredensial akun ini saat meminta otorisasi di langkah berikutnya.

Setelah Anda membuat VPC, Anda dapat menemukan ID VPC di konsol VPC Amazon dengan melihat. VPCs

3. Otorisasi VPC mengintip dengan non-Amazon GameLift Servers sumber daya.

Saat Amazon GameLift Servers menciptakan armada baru dan VPC yang sesuai, itu juga mengirimkan permintaan untuk mengintip dengan VPC untuk non-Amazon GameLift Servers sumber daya. Anda harus melakukan pra-otorisasi permintaan tersebut. Langkah ini memperbarui grup keamanan untuk VPC Anda.

Menggunakan kredensi akun yang mengelola non-Amazon GameLift Servers sumber daya, panggil layanan API [CreateVpcPeeringAuthorization\(\)](#) atau gunakan perintah AWS CLI. `create-vpc-peering-authorization` Tentukan informasi berikut:

- ID VPC Peer — ID VPC dengan non-Amazon GameLift Servers sumber daya.
- Amazon GameLift Servers AWS ID akun — ID akun yang Anda gunakan untuk mengelola Amazon GameLift Servers armada.

Setelah Anda mengesahkan peering VPC, otorisasi tetap berlaku selama 24 jam kecuali dicabut. Anda dapat mengelola otorisasi peering VPC menggunakan operasi berikut:

- [DescribeVpcPeeringAuthorizations\(\)](#) (AWS CLI `describe-vpc-peering-authorizations`).
- [DeleteVpcPeeringAuthorization\(\)](#) (AWS CLI `delete-vpc-peering-authorization`).

4. Ikuti instruksi untuk [membuat armada baru menggunakan AWS CLI](#). Sertakan parameter tambahan berikut:

- `peer-vpc-aws-account-id` — ID untuk akun yang Anda gunakan untuk mengelola VPC dengan non-Amazon GameLift Servers sumber daya.
- `peer-vpc-id`— ID VPC dengan non-Amazon GameLift Servers akun.

Panggilan yang berhasil ke [create-fleet](#) dengan parameter peering VPC menghasilkan armada baru dan permintaan peering VPC baru. Status armada diatur ke Baru dan proses aktivasi armada dimulakan. Status permintaan koneksi peering diatur ke `initiating-request`. Anda dapat melacak keberhasilan atau kegagalan permintaan peering dengan menelepon [describe-vpc-peering-connections](#).

Saat meminta armada baru dan koneksi peering VPC, kedua tindakan tersebut bisa berhasil atau gagal. Jika armada gagal selama proses pembuatan, koneksi peering VPC tidak akan dibuat. Demikian juga, jika koneksi peering VPC gagal karena alasan apa pun, armada baru akan gagal berpindah dari status Mengaktifkan ke Aktif.

Note

Koneksi peering VPC baru tidak selesai sampai armada untuk aktif. Artinya koneksi tidak tersedia dan tidak dapat digunakan selama proses instalasi bangunan server game.

Contoh berikut membuat armada baru dan koneksi peering antara VPC yang telah dibuat sebelumnya dan VPC untuk armada baru. VPC yang telah ditetapkan sebelumnya diidentifikasi secara unik dengan kombinasi non-Amazon GameLift Servers AWS ID akun dan ID VPC.

```
$ AWS gamelift create-fleet
  --name "My_Fleet_1"
  --description "The sample test fleet"
  --ec2-instance-type "c5.large"
  --fleet-type "ON_DEMAND"
  --build-id "build-1111aaaa-22bb-33cc-44dd-5555eeee66ff"
  --runtime-configuration "GameSessionActivationTimeoutSeconds=300,
                           MaxConcurrentGameSessionActivations=2,
                           ServerProcesses=[{LaunchPath=C:\game\Bin64.dedicated
\MultiplayerSampleProjectLauncher_Server.exe,
                                           Parameters=+sv_port 33435 +start_lobby,
                                           ConcurrentExecutions=10}]"
  --new-game-session-protection-policy "FullProtection"
  --resource-creation-limit-policy "NewGameSessionsPerCreator=3,
```

```

PolicyPeriodInMinutes=15"

--ec2-inbound-permissions
"FromPort=33435,ToPort=33435,IpRange=0.0.0.0/0,Protocol=UDP"

"FromPort=33235,ToPort=33235,IpRange=0.0.0.0/0,Protocol=UDP"
--metric-groups "EMEAfleets"
--peer-vpc-aws-account-id "111122223333"
--peer-vpc-id "vpc-a11a11a"

```

Versi yang dapat disalin:

```

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"

```

Memecahkan masalah peering VPC

Jika Anda mengalami kesulitan membuat koneksi peering VPC untuk Anda Amazon GameLift Servers server game, pertimbangkan akar penyebab umum ini:

- Otorisasi untuk koneksi yang diminta tidak ditemukan:
 - Periksa status otorisasi VPC untuk non-Amazon GameLift Servers VPC. Mungkin tidak ada atau mungkin sudah kedaluwarsa.
 - Periksa wilayah dari dua yang VPCs Anda coba peer. Jika mereka tidak berada di wilayah yang sama, mereka tidak dapat di-peer.
- Blok CIDR (lihat [Konfigurasi koneksi peering VPC yang tidak valid](#)) dari keduanya tumpang tindih. VPCs Blok IPv4 CIDR yang ditugaskan ke peered tidak VPCs dapat tumpang tindih. Blok CIDR dari VPC untuk Anda Amazon GameLift Servers armada secara otomatis ditetapkan dan tidak dapat diubah, jadi Anda harus mengubah blok CIDR untuk VPC untuk non-Amazon GameLift Servers sumber daya. Untuk menyelesaikan masalah ini:

- Cari blok CIDR ini untuk Anda Amazon GameLift Servers armada dengan `DescribeVpcPeeringConnections()`.
- Buka konsol VPC Amazon, temukan VPC untuk non-Amazon GameLift Servers sumber daya, dan ubah blok CIDR sehingga tidak tumpang tindih.
- Armada baru tidak aktif (saat meminta peering VPC dengan armada baru). Jika armada baru gagal menjadi Aktif, tidak ada VPC untuk di-peer, sehingga koneksi peering tidak berhasil.

Amazon GameLift Servers armada kontainer terkelola

Amazon GameLift Servers armada kontainer terkelola menawarkan platform berbasis cloud untuk menghosting perangkat lunak server game kontainer Anda. Dengan armada kontainer, Anda mendapatkan fleksibilitas, keamanan, dan keandalan AWS Cloud sumber daya, yang selanjutnya dioptimalkan untuk hosting game multipemain. Amazon GameLift Servers Layanan ini menyediakan perkakas manajemen host yang kuat.

 Percepat orientasi dengan alat ini untuk kontainer terkelola:

- [Kit starter kontainer](#) merampingkan integrasi dan pengaturan armada. Ini menambahkan fitur manajemen sesi permainan penting ke server game Anda, dan menggunakan templat yang telah dikonfigurasi sebelumnya untuk membangun armada kontainer dan pipeline penyebaran otomatis untuk server game Anda. Setelah penerapan, gunakan Amazon GameLift Servers konsol dan alat API untuk memantau kinerja armada, mengelola sesi game, dan menganalisis metrik.
- Untuk pengembang Unreal Engine atau Unity, gunakan [Amazon GameLift Servers plugin dan server mesin game SDKs untuk mengintegrasikan server](#) game Anda dan membangun armada kontainer dari dalam lingkungan pengembangan mesin game Anda. Alur kerja terpandu plugin membantu Anda membuat solusi yang cepat dan sederhana dengan hosting berbasis cloud menggunakan kontainer terkelola. Anda dapat membangun fondasi ini untuk membuat solusi hosting khusus untuk game Anda.

Armada kontainer terkelola terdiri dari serangkaian komputasi virtual yang Amazon GameLift Servers memiliki dan beroperasi atas nama Anda dan berdasarkan pilihan konfigurasi Anda. Komputasi adalah instans Amazon Elastic Compute Cloud EC2 (Amazon) dengan Amazon GameLift Servers sistem operasi Linux. Instans secara fisik terletak di Wilayah AWS atau Local Zones. Saat membuat

armada kontainer, Anda memilih jenis EC2 instans untuk komputasi berdasarkan daya komputasi, memori, penyimpanan, kemampuan jaringan, dan faktor lainnya.

Untuk armada kontainer terkelola, Anda menyimpan gambar kontainer berbasis Linux di repositori Amazon Elastic Container Registry (Amazon ECR) Registry (Amazon Elastic Container Registry) dan membuat definisi grup kontainer untuk menggambarkan arsitektur kontainer Anda. Saat Anda membuat armada, Amazon GameLift Servers layanan menggunakan definisi grup kontainer untuk menyebarkan gambar kontainer Anda ke instance armada. Saat kontainer meluncurkan proses server game, setiap proses membuat koneksi ke Amazon GameLift Servers layanan dan melaporkan kesiapan untuk meng-host sesi game.

Selain penerapan armada, Amazon GameLift Servers menangani tugas manajemen host berikut sehingga Anda tidak perlu:

- Melacak status semua kontainer di armada dan menggantikan yang basi atau tidak sehat.
- Menangani otentikasi untuk komunikasi antara proses server dan Amazon GameLift Servers layanan.
- Menawarkan alat auto-scaling yang menyesuaikan kapasitas armada secara dinamis untuk memenuhi permintaan pemain.
- Melaporkan metrik kinerja untuk EC2 instance armada, kontainer, dan proses server.

Lihat topik berikut tentang cara menyiapkan dan memelihara armada kontainer terkelola:

- [Peta jalan pengembangan untuk hosting dengan kontainer Amazon GameLift Servers terkelola](#)
- [Buat sebuah Amazon GameLift Servers armada kontainer terkelola](#)
- [Sesuaikan armada Amazon GameLift Servers kontainer](#)
- [Menskalakan kapasitas hosting game dengan Amazon GameLift Servers](#)
- [Perbarui Amazon GameLift Servers armada kontainer terkelola](#)

Buat sebuah Amazon GameLift Servers armada kontainer terkelola

Buat sebuah Amazon GameLift Servers armada kontainer terkelola untuk menyebarkan dan meng-host server game kontainer Anda di Cloud. AWS Saat membuat armada kontainer, Anda dapat menentukan definisi grup kontainer dengan pengaturan konfigurasi, termasuk satu atau beberapa gambar kontainer, termasuk gambar dengan server game Anda. Anda juga dapat memilih untuk

membuat armada kontainer kosong dan menambahkan atau memperbarui definisi grup kontainer armada nanti.

Anda dapat menggunakan [Amazon GameLift Servers konsol](#) atau AWS Command Line Interface (AWS CLI) untuk membuat armada kontainer.

Setelah Anda membuat armada kontainer baru, status armada melewati beberapa tahap sebagai Amazon GameLift Servers menyebarkan grup kontainer Anda ke setiap instance armada dan memulai server game. Ketika armada mencapai status **ACTIVE**, itu siap untuk menjadi tuan rumah sesi permainan. Jika Anda membuat armada tanpa definisi grup kontainer, armada tidak akan mencapai status aktif. Untuk bantuan terkait masalah pembuatan armada, lihat [Masalah Amazon GameLift Servers armada debug](#).

Console

Di [Amazon GameLift Servers konsol](#), pilih Wilayah AWS tempat Anda ingin membuat armada. Definisi grup kontainer harus berada di wilayah yang sama di mana Anda ingin membuat armada.

Buka bilah navigasi kiri konsol dan pilih Kontainer terkelola: Armada. Pada halaman Armada, pilih **Buat armada kontainer**.

Langkah 1: Tentukan detail armada kontainer terkelola

1. Di bagian Detail armada kontainer, masukkan deskripsi armada.
2. Tentukan peran IAM untuk armada. Peran ini memiliki izin yang Amazon GameLift Servers harus mengelola armada kontainer atas nama Anda. Untuk bantuan membuat peran layanan yang diperlukan, lihat [Menyiapkan peran layanan IAM untuk Amazon GameLift Servers](#).
3. Pilih opsi konfigurasi Log. CloudWatch Opsi dipilih secara default. Berikan informasi yang diperlukan berdasarkan opsi yang Anda pilih.
4. Tambahkan grup kontainer ke armada. Ini adalah langkah opsional. Anda dapat memilih untuk membuat armada tanpa grup kontainer dengan rencana untuk menambahkannya nanti. Armada tanpa grup kontainer tidak akan menyebarkan instance armada apa pun dan belum dapat menghosting game apa pun, tetapi sumber daya armada dibuat.
 - Pilih definisi grup kontainer server game. Secara opsional tentukan versi definisi yang ingin Anda terapkan. Jika Anda tidak menentukan nomor versi, Amazon GameLift Servers secara otomatis menggunakan versi terbaru.

- Secara opsional tambahkan definisi dan versi grup kontainer per-instance. Jika Anda tidak menentukan nomor versi, Amazon GameLift Servers secara otomatis menggunakan versi terbaru.
5. Dalam Detail tambahan, Anda dapat mengatur beberapa penyesuaian opsional. Tak satu pun dari pengaturan ini diperlukan untuk membuat armada kontainer.

Langkah 2: Tentukan detail contoh

1. Dalam penyebaran Instance, pilih satu atau beberapa lokasi jarak jauh untuk menerapkan instance. Wilayah asal dipilih secara otomatis (ini adalah Wilayah tempat Anda membuat armada). Jika Anda memilih lokasi tambahan, instans armada juga digunakan di lokasi tersebut.

Important

Untuk menggunakan Wilayah yang tidak diaktifkan secara default, aktifkan di wilayah Anda Akun AWS.

- Armada dengan Wilayah yang tidak diaktifkan yang Anda buat sebelum 28 Februari 2022 tidak terpengaruh.
- Untuk membuat armada multi-lokasi baru atau memperbarui armada multi-lokasi yang ada, pertama-tama aktifkan Wilayah apa pun yang Anda pilih untuk digunakan.

Untuk informasi selengkapnya tentang Wilayah yang tidak diaktifkan secara default dan cara mengaktifkannya, lihat [Mengelola Wilayah AWS](#) di Referensi Umum AWS.

2. Pilih konfigurasi Instance untuk armada. Konsol secara otomatis menghitung vCPU minimum dan memori yang diperlukan (berdasarkan batas total yang Anda tetapkan untuk setiap grup kontainer). Ini menyaring daftar lengkap jenis instans yang tersedia berdasarkan persyaratan sumber daya dan lokasi yang Anda masukkan. Anda dapat menambahkan filter tambahan sesuai kebutuhan.

Untuk informasi selengkapnya tentang memilih jenis instans, lihat [Konfigurasi armada kontainer](#). Ukuran jenis instance yang Anda pilih akan memengaruhi bagaimana grup kontainer server game dikemas ke setiap instance armada. Bergantung pada pilihan Anda,

pertimbangkan untuk meninjau pengaturan Anda untuk grup kontainer server game yang diinginkan per instance.

Langkah 4: Tinjau dan buat

- Tinjau pengaturan konfigurasi armada Anda.

Anda dapat memperbarui metadata dan konfigurasi armada kapan saja, terlepas dari status armada. Untuk informasi selengkapnya, lihat [Perbarui konfigurasi Amazon GameLift Servers armada](#). Anda dapat memperbarui kapasitas armada setelah armada mencapai status AKTIF. Untuk informasi selengkapnya, lihat [Menskalakan kapasitas hosting game dengan Amazon GameLift Servers](#). Anda juga dapat menambahkan atau menghapus lokasi jarak jauh.

Setelah selesai meninjau, pilih Buat.

Jika permintaan Anda berhasil, konsol akan menampilkan halaman detail untuk sumber daya armada baru. Awalnya statusnya adalah NEW, sebagai Amazon GameLift Servers memulai proses pembuatan armada. Anda dapat melacak status armada baru di halaman Armada. Armada siap menjadi tuan rumah sesi permainan ketika mencapai status ACTIVE.

AWS CLI

Untuk membuat armada kontainer dengan AWS CLI, buka jendela baris perintah dan gunakan `create-container-fleet` perintah. Untuk informasi selengkapnya tentang perintah ini, lihat [create-container-fleet](#) di Referensi AWS CLI Perintah.

Contoh `create-container-fleet` permintaan yang ditunjukkan di bawah ini membuat armada kontainer baru dengan karakteristik sebagai berikut:

- `ContainerGroupsConfiguration` Menentukan definisi grup kontainer server game saja: `MyAdventureGameContainerGroup`. Jumlah grup kontainer server game yang akan dikerahkan ke setiap instance armada dihitung oleh Amazon GameLift Servers.
- Armada menggunakan `c5.large` instans On-Demand secara default.
- Secara default, armada membuka satu set port koneksi dan port izin masuk seperti yang dihitung oleh Amazon GameLift Servers. Ini menyebarkan grup kontainer ke lokasi berikut:

```
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" \  

```

Jika permintaan `create-fleet` berhasil, Amazon GameLift Servers mengembalikan satu set atribut armada yang mencakup pengaturan konfigurasi yang Anda minta dan ID armada kontainer baru. Amazon GameLift Servers kemudian menetapkan status armada dan status lokasi ke `New` dan memulai proses aktivasi armada. Anda dapat melacak status armada dan melihat informasi armada lainnya menggunakan perintah CLI ini:

- [describe-fleet-events](#)
- [describe-container-fleet](#)
- [describe-fleet-capacity](#)
- [describe-fleet-port-settings](#)
- [describe-fleet-utilization](#)
- [describe-fleet-location-capacity](#)
- [describe-fleet-location-utilization](#)

Anda dapat mengubah kapasitas armada dan pengaturan konfigurasi lainnya sesuai kebutuhan menggunakan perintah berikut:

- [update-container-fleet](#)
- [update-fleet-capacity](#)
- [update-fleet-port-settings](#)
- [create-fleet-locations](#)
- [delete-fleet-locations](#)

Perbarui Amazon GameLift Servers armada kontainer terkelola

Anda dapat memperbarui sebagian besar properti armada kontainer terkelola, termasuk definisi grup kontainer. Bergantung pada pengaturan yang diperbarui, pembaruan armada mungkin memulai penyebaran armada baru. Dalam penyebaran armada, semua instance dalam armada dihapus dan diganti dengan instance dengan konfigurasi baru. Pengaturan yang memerlukan penerapan meliputi:

- Definisi grup kontainer, termasuk pembaruan pada gambar kontainer

- Rentang port koneksi dan izin masuk
- Konfigurasi log

Anda dapat melacak status penyebaran armada di [Amazon GameLift Servers konsol](#) atau AWS Command Line Interface (AWS CLI) untuk membuat armada kontainer.

Console

Di [Amazon GameLift Servers konsol](#), pilih Wilayah AWS tempat Anda ingin membuat armada. Definisi grup kontainer harus berada di wilayah yang sama di mana Anda ingin membuat armada.

Buka bilah navigasi kiri konsol dan pilih Kontainer terkelola: Armada. Pada halaman armada kontainer terkelola, pilih armada dari daftar dan pilih Edit.

1. Perbarui pengaturan armada kontainer sesuai kebutuhan. Setelah selesai, pilih Create (Buat).
2. Jika pembaruan Anda memerlukan penerapan armada, Anda diminta untuk menentukan opsi penerapan sebagai berikut:
 - Perlindungan sesi permainan. Anda dapat memilih untuk melindungi instance armada yang memiliki sesi permainan aktif (penerapan aman). Dengan pengaturan ini, instance armada tidak diganti sampai setelah sesi permainan berakhir. Atau, Anda dapat memilih untuk mengganti instance armada terlepas dari aktivitas sesi game (penerapan tidak aman). Penerapan yang tidak aman berguna selama fase pengembangan dan pengujian untuk mengurangi waktu penerapan.
 - Persentase sehat minimum. Anda dapat mengatur seberapa cepat instance armada diganti. Gunakan pengaturan ini untuk mempertahankan jumlah minimal tugas sehat selama penerapan. Nilai rendah memprioritaskan kecepatan penerapan, sementara nilai tinggi memastikan bahwa ketersediaan server game tetap tinggi selama penerapan.
 - Strategi kegagalan penerapan. Tentukan tindakan apa yang harus diambil jika penerapan gagal. Kegagalan penerapan berarti bahwa beberapa kontainer yang diperbarui telah gagal memeriksa status dan dianggap terganggu. Anda dapat mengatur penerapan untuk secara otomatis memutar kembali semua instance armada ke status yang digunakan sebelumnya. Atau Anda dapat memilih untuk mempertahankan beberapa instance armada yang terganggu untuk digunakan dalam debugging.

Jika permintaan Anda berhasil, konsol akan menampilkan tab Deployment untuk armada kontainer terkelola. Gunakan tab ini untuk melacak status setiap penerapan. Jika Anda memulai

penyebaran baru untuk armada, tindakan ini secara otomatis membatalkan penerapan apa pun yang saat ini sedang dalam proses untuk armada.

AWS CLI

Untuk membuat armada kontainer dengan AWS CLI, buka jendela baris perintah dan gunakan `update-container-fleet` perintah. Untuk informasi selengkapnya tentang perintah ini, lihat [update-container-fleet](#) di Referensi AWS CLI Perintah.

Contoh berikut memperbarui armada kontainer yang ada dengan karakteristik sebagai berikut:

- Ini memperbarui definisi grup kontainer server game untuk menggunakan versi 2.
- Ini menentukan opsi penyebaran yang aman.

```
{
  "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"
}
```

Sesuaikan armada Amazon GameLift Servers kontainer

Topik di bagian ini menjelaskan beberapa fitur opsional untuk kontainer Amazon GameLift Servers terkelola. Anda dapat memilih untuk menggunakan salah satu atau semua fitur ini.

Topik

- [Tetapkan batas sumber daya](#)
- [Tentukan wadah penting](#)
- [Konfigurasi koneksi jaringan](#)
- [Siapkan pemeriksaan kesehatan untuk wadah](#)
- [Tetapkan dependensi wadah](#)

- [Konfigurasi armada kontainer](#)

Tetapkan batas sumber daya

Untuk setiap grup kontainer, Anda dapat menentukan berapa banyak memori dan daya komputasi yang dibutuhkan grup kontainer untuk menjalankan perangkat lunak. Amazon GameLift Servers bergantung pada informasi ini untuk mengelola sumber daya di seluruh grup kontainer. Ini juga menggunakan informasi ini untuk menghitung berapa banyak grup kontainer server game yang dapat dipegang oleh gambar armada. Anda juga dapat menetapkan batas untuk wadah individu.

Anda dapat menetapkan batas maksimum pada memori dan daya komputasi untuk grup kontainer. Secara default, sumber daya ini dibagikan oleh semua kontainer dalam grup. Anda dapat menyesuaikan manajemen sumber daya lebih lanjut dengan menetapkan batas untuk setiap kontainer.

Tetapkan batas opsional untuk wadah individu

Menyetel batas sumber daya khusus kontainer memungkinkan Anda memberikan kontrol yang lebih besar atas bagaimana kontainer individu dapat menggunakan sumber daya grup. Jika Anda tidak menetapkan batas khusus kontainer, semua kontainer dalam grup berbagi sumber daya grup. Berbagi menawarkan fleksibilitas yang lebih besar untuk menggunakan sumber daya di tempat yang dibutuhkan. Ini juga meningkatkan potensi proses untuk bersaing satu sama lain dan mengakibatkan kegagalan kontainer.

Tetapkan salah satu `ContainerDefinition` properti berikut untuk wadah apa pun.

- `MemoryHardLimitMebibytes`— Tetapkan batas memori maksimum untuk wadah. Jika wadah melebihi batas ini, itu menghasilkan restart.
- `Vcpulimit` — Cadangan jumlah minimum sumber daya vCPU untuk penggunaan eksklusif kontainer. Wadah selalu memiliki jumlah cadangan yang tersedia untuk itu. Ini dapat melebihi minimum ini kapan saja, jika sumber daya tambahan tersedia. (1024 unit CPU setara dengan 1 vCPU.)

Tetapkan batas sumber daya total untuk grup kontainer

Jika Anda menetapkan batas untuk kontainer individual, Anda mungkin perlu memodifikasi berapa banyak memori dan sumber daya vCPU yang dibutuhkan grup kontainer. Tujuannya adalah untuk mengalokasikan sumber daya yang cukup untuk mengoptimalkan kinerja server game. Amazon GameLift Servers menggunakan batasan ini untuk menghitung cara mengemas grup

kontainer server game pada instance armada. Anda juga akan menggunakannya saat memilih jenis instance untuk armada kontainer.

Hitung total memori dan vCPU yang dibutuhkan untuk grup kontainer. Pertimbangkan hal berikut:

- Apa saja proses yang berjalan di semua kontainer dalam grup kontainer? Tambahkan sumber daya yang diperlukan untuk proses ini. Catat batasan khusus wadah apa pun.
- Berapa banyak proses server game bersamaan yang Anda rencanakan untuk dijalankan di setiap grup kontainer? Anda menentukan ini dalam gambar wadah server game Anda.

Berdasarkan perkiraan persyaratan grup kontainer Anda, tetapkan

`ContainerGroupDefinition` properti berikut:

- `TotalMemoryLimitMebibytes`— Tetapkan batas memori maksimum untuk grup kontainer. Semua kontainer dalam grup berbagi memori yang dialokasikan. Jika Anda menetapkan batas kontainer individual, batas memori total harus sama dengan atau lebih besar dari batas memori spesifik kontainer tertinggi.
- `TotalVcpuLimit`— Tetapkan batas vCPU maksimum untuk grup kontainer. Semua kontainer dalam grup berbagi sumber daya CPU yang dialokasikan. Jika Anda menetapkan batas kontainer individual, batas total CPU harus sama dengan atau lebih besar dari jumlah semua batas CPU khusus kontainer. Sebagai praktik terbaik, pertimbangkan untuk mengatur nilai ini untuk menggandakan jumlah batas CPU kontainer.

Contoh skenario

Katakanlah kita mendefinisikan grup kontainer server game dengan tiga kontainer berikut:

- Container A adalah wadah server game kami. Kami memperkirakan kebutuhan sumber daya untuk satu server game di 512 MiB dan 1024 CPU. Kami berencana agar container menjalankan 1 proses server. Karena wadah ini menjalankan perangkat lunak kami yang paling penting, kami tidak menetapkan batas memori atau batas cadangan vCPU.
- Container B run adalah wadah pendukung dengan persyaratan sumber daya diperkirakan 1024 MiB dan 1536 CPU. Kami menetapkan batas memori 2048 MiB, dan batas cadangan CPU 1024 CPU.
- Container C adalah wadah pendukung lainnya. Kami menetapkan batas memori keras 512 MiB dan batas cadangan CPU 512 CPU.

Dengan menggunakan informasi ini, kami menetapkan batas total berikut untuk grup kontainer:

- Total batas memori: 7680 MiB. Nilai ini melebihi batas memori tertinggi (1024 MiB).
- Total batas CPU: 13312 CPU. Nilai ini melebihi jumlah batas CPU (1024+512 CPU).

Tentukan wadah penting

Untuk grup kontainer per-instance, tetapkan setiap kontainer sebagai esensial atau non-esensial. Grup kontainer per instance harus memiliki setidaknya satu wadah pendukung penting. Wadah penting melakukan pekerjaan kritis dari kelompok kontainer. Wadah penting selalu diharapkan berjalan. Jika gagal, seluruh grup kontainer akan dimulai ulang.

Setel `ContainerDefinition` properti `Essential` ke `true` atau `false` untuk setiap kontainer.

Konfigurasi koneksi jaringan

Anda dapat menyesuaikan akses jaringan agar lalu lintas eksternal terhubung ke kontainer apa pun dalam armada kontainer. Misalnya, Anda harus membuat koneksi jaringan ke wadah yang menjalankan proses server game Anda, sehingga klien game dapat bergabung dan memainkan game Anda. Klien game terhubung ke server game menggunakan port dan alamat IP.

Dalam armada kontainer, koneksi antara klien dan server tidak langsung. Secara internal, proses dalam wadah mendengarkan pada port kontainer. Secara eksternal, lalu lintas masuk terhubung ke instance armada menggunakan port koneksi. Amazon GameLift Servers memelihara pemetaan antara port kontainer internal dan port koneksi yang menghadap ke luar, sehingga lalu lintas yang masuk diarahkan ke proses yang benar pada instance.

Amazon GameLift Servers menyediakan lapisan kontrol ekstra untuk koneksi jaringan Anda. Setiap armada kontainer memiliki pengaturan izin masuk, yang memungkinkan Anda mengontrol akses ke setiap port koneksi yang menghadap ke luar. Misalnya, Anda dapat menghapus izin untuk semua port koneksi untuk mematikan semua akses ke kontainer armada.

Anda dapat memperbarui izin masuk armada, port koneksi, dan port kontainer.

Atur rentang port kontainer

Konfigurasi rentang port kontainer sebagai bagian dari setiap definisi kontainer. Ini adalah parameter yang diperlukan untuk definisi grup kontainer. Anda perlu mengkonfigurasi port yang cukup untuk mengakomodasi semua proses yang berjalan secara bersamaan yang membutuhkan akses eksternal. Beberapa kontainer tidak membutuhkan port apa pun.

Wadah server game Anda, yang menjalankan server game Anda, membutuhkan port untuk setiap proses server game yang berjalan secara bersamaan. Proses server game mendengarkan pada port yang ditetapkan dan melaporkannya ke Amazon GameLift Servers.

Atur rentang port koneksi

Konfigurasi armada kontainer Anda dengan satu set port koneksi. Port koneksi menyediakan akses eksternal ke instance armada yang menjalankan kontainer Anda. Amazon GameLift Servers menetapkan port koneksi dan memetakannya ke port kontainer sesuai kebutuhan.

Secara default, Amazon GameLift Servers menghitung jumlah port yang diperlukan untuk semua grup kontainer dan menetapkan rentang port untuk mengakomodasi mereka. Kami sangat menyarankan Anda menggunakan nilai Amazon GameLift Servers terhitung, yang diperbarui saat Anda menerapkan pembaruan ke definisi grup kontainer. Jika Anda perlu menyesuaikan rentang port koneksi, gunakan panduan berikut.

Saat Anda membuat armada kontainer, tentukan rentang port koneksi (lihat [ContainerFleet: InstanceConnectionPortRange](#)). Pastikan rentang memiliki port yang cukup untuk dipetakan ke setiap port kontainer yang ditentukan di semua kontainer di kedua grup kontainer di armada. Untuk menghitung port koneksi minimum yang diperlukan, gunakan rumus berikut:

```
[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]
```

Sebagai praktik terbaik, gandakan jumlah minimum port koneksi.

Note

Jumlah port koneksi berpotensi membatasi jumlah grup kontainer server game per instance. Jika armada hanya memiliki port koneksi yang cukup untuk satu grup kontainer server game per instance, Amazon GameLift Servers akan menyebarkan hanya satu grup kontainer server game, bahkan jika instance memiliki daya komputasi yang cukup untuk beberapa grup kontainer server game.

Tetapkan izin masuk

Izin masuk mengontrol akses eksternal ke armada kontainer dengan menentukan port koneksi mana yang akan dibuka untuk lalu lintas masuk. Anda dapat menggunakan pengaturan ini untuk mengaktifkan dan menonaktifkan akses jaringan armada sesuai kebutuhan.

Secara default, Amazon GameLift Servers menghitung jumlah port yang diperlukan untuk semua grup kontainer dan menetapkan rentang port untuk mengakomodasi mereka. Kami sangat menyarankan Anda menggunakan nilai Amazon GameLift Servers terhitung, yang diperbarui saat Anda menerapkan pembaruan ke definisi grup kontainer. Jika Anda perlu menyesuaikan rentang port koneksi, gunakan panduan berikut.

Saat Anda membuat armada kontainer, tentukan satu set izin masuk (lihat [ContainerFleet: InstanceInboundPermissions](#)). Port izin masuk harus sesuai dengan rentang port koneksi armada.

Contoh skenario

Contoh ini menggambarkan cara mengatur ketiga properti koneksi jaringan.

- Grup kontainer server game armada kami memiliki 1 kontainer, yang menjalankan 1 proses server game.

Dalam definisi grup kontainer server game, kami menetapkan `PortConfiguration` parameter untuk wadah ini sebagai berikut:

```
"PortConfiguration": {  
  "ContainerPortRanges": [ { "FromPort": 10, "ToPort": 20, "Protocol": "TCP"} ]  
}
```

- Armada kami juga memiliki grup kontainer per instance dengan 1 kontainer. Ini memiliki 1 proses yang membutuhkan akses jaringan. Dalam definisi kontainer per-instance, kami menetapkan `PortConfiguration` parameter untuk wadah ini sebagai berikut:

```
"PortConfiguration": {  
  "ContainerPortRanges": [ { "FromPort": 25, "ToPort": 25, "Protocol": "TCP"} ] }
```

- Armada kami dikonfigurasi dengan 20 grup kontainer server game per instance armada. Dengan informasi ini, kita dapat menggunakan rumus untuk menghitung jumlah port koneksi yang kita butuhkan:
 - Minimum: 21 port [1 port kontainer server game * 20 grup kontainer server game per instance +1 port kontainer per instance]
 - Praktik terbaik: 42 port [port minimum* 2]

Saat membuat armada kontainer, kami mengatur `ConnectionPortRange` parameter sebagai berikut:

```
"ConnectionPortRange": { "FromPort": 1010, "ToPort": 1071 }
```

- Kami ingin mengizinkan akses ke semua port koneksi yang tersedia. Saat membuat armada kontainer, kami mengatur InstanceInboundPermissions parameter sebagai berikut:

```
"InstanceInboundPermissions": [  
  {"FromPort": 1010, "ToPort": 1071, "IpRange": "10.24.34.0/23", "Protocol":  
    "TCP"} ]
```

Siapkan pemeriksaan kesehatan untuk wadah

Sebuah kontainer secara otomatis restart jika mengalami kegagalan terminal dan berhenti berjalan. Jika kontainer dianggap penting, ia meminta seluruh grup kontainer untuk memulai ulang.

Semua wadah server game secara otomatis dianggap penting. Support container dapat ditunjuk penting, tetapi mereka harus memiliki mekanisme untuk melaporkan kesehatan. Anda juga dapat mengatur pemeriksaan kesehatan untuk wadah pendukung yang tidak penting.

Anda dapat menentukan kriteria khusus tambahan untuk mengukur kesehatan kontainer dan menggunakan pemeriksaan kesehatan untuk menguji kriteria tersebut. Untuk menyiapkan pemeriksaan kesehatan kontainer, Anda dapat mendefinisikannya dalam gambar kontainer Docker atau dalam definisi kontainer Anda. Jika Anda menetapkan pemeriksaan kesehatan dalam definisi kontainer, itu akan mengganti pengaturan apa pun dalam gambar kontainer.

Tetapkan SupportContainerDefinition properti berikut untuk pemeriksaan kesehatan kontainer:

- **Command**— Berikan perintah yang memeriksa beberapa aspek kesehatan wadah. Anda memutuskan kriteria apa yang akan digunakan untuk mengukur kesehatan. Perintah harus menghasilkan nilai keluar 1 (tidak sehat) atau 0 (sehat).
- **StartPeriod**— Tentukan penundaan awal sebelum kegagalan pemeriksaan kesehatan mulai menghitung. Penundaan ini memberi waktu wadah untuk mem-bootstrap prosesnya.
- **Interval**— Putuskan seberapa sering menjalankan perintah pemeriksaan kesehatan. Seberapa cepat Anda ingin mendeteksi dan menyelesaikan kegagalan kontainer?
- **Timeout**— Putuskan berapa lama menunggu keberhasilan atau kegagalan sebelum mencoba kembali perintah pemeriksaan kesehatan. Berapa lama perintah pemeriksaan kesehatan harus diselesaikan?

- **Retries**— Berapa kali perintah pemeriksaan kesehatan harus dicoba lagi sebelum mendaftarkan kegagalan?

Tetapkan dependensi wadah

Dalam setiap grup kontainer Anda dapat mengatur dependensi antar kontainer berdasarkan status kontainer. Ketergantungan berdampak ketika kontainer dependen dapat memulai atau mematikan berdasarkan status wadah lain.

Kasus penggunaan utama untuk dependensi adalah membuat urutan startup dan shutdown untuk grup kontainer.

Misalnya, Anda mungkin ingin Container A dimulai terlebih dahulu dan berhasil diselesaikan sebelum Container B dan C dimulai. Untuk mencapai hal ini, pertama buat dependensi untuk Container B pada Container A, dengan syarat bahwa Container A harus berhasil diselesaikan. Kemudian buat dependensi untuk Container C pada Container A dengan kondisi yang sama. Urutan startup terjadi dalam urutan terbalik untuk shutdown.

Konfigurasi armada kontainer

Saat Anda membuat armada kontainer, pertimbangkan poin keputusan berikut. Sebagian besar poin ini bergantung pada arsitektur dan konfigurasi kontainer Anda.

Tentukan di mana Anda ingin menyebarkan armada Anda

Secara umum, Anda ingin menyebarkan armada Anda secara geografis di dekat pemain Anda untuk meminimalkan latensi. Anda dapat menyebarkan armada kontainer Anda ke Masing-masing Wilayah AWS yang Amazon GameLift Servers mendukung. Jika Anda ingin menyebarkan server game yang sama ke lokasi geografis tambahan, Anda dapat menambahkan lokasi terpencil ke armada termasuk Wilayah AWS dan Local Zones. Untuk armada multi-lokasi, Anda dapat menyesuaikan kapasitas secara mandiri di setiap lokasi armada. Untuk informasi selengkapnya tentang lokasi armada yang didukung, lihat [Amazon GameLift Servers lokasi layanan](#).

Pilih jenis dan ukuran instans untuk armada Anda

Amazon GameLift Servers mendukung berbagai jenis EC2 instans Amazon, yang semuanya tersedia untuk digunakan dengan armada kontainer. Ketersediaan dan harga jenis instans bervariasi menurut lokasi. Anda dapat melihat daftar jenis instans yang didukung, yang difilter berdasarkan lokasi, di Amazon GameLift Servers konsol (di bawah Resources, Instance, dan kuota layanan).

Saat memilih jenis instance, pertama-tama pertimbangkan keluarga instance. Keluarga instans menawarkan berbagai kombinasi kemampuan CPU, memori, penyimpanan, dan jaringan. Dapatkan informasi lebih lanjut tentang [keluarga EC2 contoh](#). Dalam setiap keluarga Anda memiliki berbagai ukuran instans untuk dipilih. Pertimbangkan masalah berikut saat memilih ukuran instans:

- Berapa ukuran instans minimum yang dapat mendukung beban kerja Anda? Gunakan informasi ini untuk menghilangkan jenis instance apa pun yang terlalu kecil.
- Ukuran tipe instance apa yang cocok untuk arsitektur kontainer Anda? Idealnya, Anda ingin memilih ukuran yang dapat menampung beberapa salinan grup kontainer server game Anda dengan ruang terbang minimal.
- Granularitas penskalaan apa yang masuk akal untuk game Anda? Kapasitas armada skala melibatkan penambahan atau penghapusan instance, dan setiap instance mewakili kemampuan untuk menyelenggarakan sejumlah sesi permainan tertentu. Pertimbangkan berapa banyak kapasitas yang ingin Anda tambahkan atau hapus dengan setiap instance. Jika permintaan pemain bervariasi ribuan dari menit ke menit, maka mungkin masuk akal untuk menggunakan contoh yang sangat besar yang dapat menampung ratusan atau ribuan sesi permainan. Sebaliknya, Anda mungkin lebih memilih kontrol penskalaan yang lebih halus dengan tipe instance yang lebih kecil.
- Apakah ada penghematan biaya yang tersedia berdasarkan ukuran? Anda mungkin menemukan bahwa biaya jenis instans tertentu bervariasi menurut lokasi karena ketersediaannya.

Atur pengaturan armada opsional lainnya

Anda dapat menggunakan fitur opsional berikut saat mengonfigurasi armada kontainer:

- Siapkan server game Anda untuk mengakses AWS sumber daya lain. Lihat [Berkomunikasi dengan AWS sumber daya lain dari armada Anda](#).
- Lindungi sesi permainan dengan pemain aktif agar tidak berhenti sebelum waktunya selama acara penurunan skala.
- Batasi jumlah sesi permainan yang dapat dibuat oleh satu individu di armada dalam rentang waktu terbatas.

Buat definisi grup kontainer untuk armada Amazon GameLift Servers kontainer

Definisi grup kontainer menjelaskan cara menerapkan aplikasi server game kontainer Anda ke armada kontainer. Ini adalah cetak biru yang memberi tahu gambar kontainer Amazon GameLift

Servers apa yang akan diterapkan ke armada dan cara menjalankannya. Saat membuat armada kontainer, Anda menentukan definisi grup kontainer yang akan diterapkan ke armada. Untuk informasi selengkapnya tentang grup kontainer, lihat [Komponen armada kontainer](#).

Sebelum Anda mulai

Kiat tentang apa yang harus dilakukan sebelum Anda mulai membuat definisi grup kontainer:

- Selesaikan gambar kontainer Anda dan dorong ke repositori Amazon Elastic Container Registry (Amazon ECR) di tempat yang sama di Wilayah AWS mana Anda berencana untuk membuat grup kontainer. Amazon GameLift Servers menangkap snapshot dari setiap gambar pada saat Anda membuat definisi grup kontainer, dan menggunakan snapshot saat menerapkan ke armada kontainer. Lihat [Membangun gambar kontainer untuk Amazon GameLift Servers](#).
- Buat definisi kontainer Anda sebagai file JSON. Definisi grup kontainer mencakup satu atau lebih definisi kontainer. Anda dapat menggunakan file JSON jika Anda membuat definisi grup kontainer menggunakan file. AWS CLI for Amazon GameLift Servers
- Verifikasi bahwa AWS pengguna Anda memiliki izin IAM untuk mengakses repositori Amazon ECR. Lihat [Contoh izin IAM untuk Amazon GameLift Servers](#).

Buat definisi grup kontainer server game

Grup kontainer server game menjalankan perangkat lunak server game Anda. Grup kontainer server game memiliki satu wadah server game, yang menjalankan server game yang dapat dieksekusi. Ini juga dapat memiliki satu atau lebih wadah dukungan untuk menjalankan perangkat lunak tambahan untuk mendukung server game Anda. (Ini kadang-kadang disebut sebagai wadah “sespan”.)

Topik ini menjelaskan cara membuat definisi grup kontainer server game sederhana menggunakan Amazon GameLift Servers konsol atau alat AWS CLI. Untuk informasi lebih rinci tentang fitur opsional, lihat [Sesuaikan armada Amazon GameLift Servers kontainer](#).

Note

Anda dapat mengubah sebagian besar definisi grup kontainer dan pengaturan definisi kontainer setelah membuatnya. Jika Anda membuat perubahan pada definisi kontainer, Amazon GameLift Servers ambil snapshot baru dari gambar kontainer yang diperbarui.

Untuk membuat definisi grup kontainer server game sederhana:

Petunjuk berikut menjelaskan cara membuat definisi grup kontainer dengan parameter minimal yang diperlukan dan menggunakan nilai Amazon GameLift Servers default.

Console

Di [Amazon GameLift Servers konsol](#), pilih Wilayah AWS tempat Anda ingin membuat grup kontainer.

Buka bilah navigasi kiri konsol dan pilih Kontainer terkelola: Definisi grup. Pada halaman definisi grup kontainer, pilih Buat definisi grup.

Langkah 1: Tentukan detail definisi grup kontainer

1. Masukkan nama definisi grup kontainer. Nama harus unik untuk Akun AWS dan Wilayah.
2. Pilih jenis grup kontainer server Game.
3. Untuk batas memori Total, masukkan sumber daya memori maksimum agar tersedia untuk semua kontainer dalam grup kontainer. Untuk bantuan menghitung nilai ini, lihat [Tetapkan batas sumber daya](#).
4. Untuk batas total vCPU, masukkan daya komputasi maksimum untuk menyediakan semua kontainer dalam grup kontainer. Untuk bantuan menghitung nilai ini, lihat [Tetapkan batas sumber daya](#).

Langkah 2: Tambahkan definisi wadah

Minimal, grup kontainer server game memiliki satu wadah server game. Di konsol, definisi kontainer pertama yang Anda buat adalah wadah server game. Langkah ini menjelaskan cara menentukan pengaturan minimum yang diperlukan untuk definisi kontainer server game.

1. Masukkan nama definisi kontainer. Setiap kontainer yang ditentukan untuk grup harus memiliki nilai nama yang unik.
2. Tautkan ke gambar kontainer dengan build server game Anda. Masukkan URI gambar Amazon ECR untuk gambar kontainer di repositori publik atau pribadi. Anda dapat menggunakan salah satu format berikut:
 - Hanya URI gambar: [Akun AWS].dkr.ecr.[Wilayah AWS].amazonaws.com/[repository ID]
 - Gambar URI + intisari: [Akun AWS].dkr.ecr.[Wilayah AWS].amazonaws.com/[repository ID]@[digest]

- Gambar URI + tag: [Akun AWS].dkr.ecr.[Wilayah AWS].amazonaws.com/[repository ID]:[tag]
3. Tentukan versi Amazon GameLift Servers Server SDK yang digunakan oleh build server game. Untuk armada kontainer, nilai ini harus 5.2.0 atau lebih besar.
 4. Dalam rentang port kontainer internal, atur protokol dan tentukan rentang port. Ukuran rentang harus lebih besar dari jumlah proses server game bersamaan yang akan berjalan di wadah ini. Jika kontainer server game hanya menjalankan satu proses server per kontainer, rentang port ini hanya membutuhkan beberapa port. Untuk detail selengkapnya, lihat [Konfigurasi koneksi jaringan](#).
 5. Tambahkan lebih banyak kontainer sesuai kebutuhan untuk menjalankan perangkat lunak pendukung tambahan. Wadah tambahan secara otomatis ditunjuk sebagai wadah pendukung. Grup kontainer server game hanya dapat memiliki satu wadah server game dan hingga delapan kontainer pendukung. Berikan pengaturan minimal yang diperlukan berikut:
 - Nama definisi kontainer
 - URI gambar ECR.
 - Port kontainer internal (Sertakan ini hanya jika kontainer memiliki proses yang membutuhkan akses jaringan.)

Langkah 3: Konfigurasi dependensi

- Jika definisi grup kontainer Anda memiliki lebih dari satu kontainer, Anda dapat mengatur dependensi antar kontainer secara opsional. Untuk informasi selengkapnya, lihat [Tetapkan dependensi wadah](#).

Langkah 3: Tinjau dan buat

1. Tinjau semua pengaturan definisi grup kontainer Anda. Gunakan Edit untuk membuat perubahan pada bagian mana pun, termasuk setiap definisi kontainer Anda untuk grup.
2. Setelah selesai meninjau, pilih Buat.

Jika permintaan Anda berhasil, konsol akan menampilkan halaman detail untuk sumber definisi grup kontainer baru. Awalnya statusnya adalah **COPYING**, karena Amazon GameLift Servers mulai mengambil snapshot dari semua gambar kontainer untuk grup. Ketika fase ini selesai, status definisi grup kontainer berubah menjadi **READY**. Definisi grup kontainer harus dalam **READY** status sebelum Anda dapat membuat armada kontainer dengannya.

AWS CLI

Saat Anda menggunakan AWS CLI untuk membuat definisi grup kontainer, pertahankan konfigurasi definisi kontainer Anda dalam file terpisah. JSON Anda dapat mereferensikan file dalam perintah CLI Anda. Lihat [Buat JSON file definisi kontainer](#) contoh skema.

Buat definisi grup kontainer

Untuk membuat definisi grup kontainer baru, gunakan perintah `create-container-group-definition` CLI. Untuk informasi selengkapnya tentang perintah ini, lihat [create-container-group-definition](#) di Referensi Perintah AWS CLI.

Contoh ini menggambarkan permintaan untuk definisi grup kontainer server game. Ini mengasumsikan bahwa Anda telah membuat file JSON dengan definisi wadah untuk grup ini.

```
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
```

Buat **JSON** file definisi kontainer

Saat Anda membuat definisi grup kontainer, Anda juga menentukan kontainer untuk grup. Definisi kontainer menentukan repositori Amazon ECR tempat image kontainer disimpan, dan konfigurasi opsional untuk port jaringan, batas penggunaan CPU dan memori, dan pengaturan lainnya.

Sebaiknya buat satu JSON file dengan konfigurasi untuk semua kontainer dalam grup kontainer. Memelihara file berguna untuk menyimpan, berbagi, versi melacak konfigurasi penting ini. Jika Anda menggunakan AWS CLI untuk membuat definisi grup kontainer Anda, Anda dapat mereferensikan file dalam perintah.

Untuk membuat definisi kontainer

1. Buat dan buka `.JSON` file baru. Misalnya:

```
[~/work/glc]$ vim SimpleServer.json
```

2. Buat definisi kontainer terpisah untuk masing-masing kontainer untuk grup. Salin konten contoh berikut dan modifikasi sesuai kebutuhan untuk wadah Anda. Untuk detail tentang sintaks definisi container, lihat [ContainerDefinitionInput](#) di Referensi Amazon GameLift Servers API.
3. Simpan file secara lokal sehingga Anda dapat merujuknya dalam perintah AWS CLI.

Contoh: Definisi wadah server game

Example

Contoh ini menjelaskan wadah penting untuk grup kontainer server game Anda. Wadah replika penting mencakup aplikasi server game Anda, Amazon GameLift Servers Agen, dan dapat menyertakan perangkat lunak pendukung lainnya untuk hosting game Anda. Definisi harus menyertakan nama, URI gambar, dan konfigurasi port. Contoh ini juga menetapkan beberapa batas sumber daya khusus kontainer.

```
{
  "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"
}
```

Memperbarui definisi grup kontainer untuk Amazon GameLift Servers armada kontainer

Anda dapat memperbarui sebagian besar properti definisi grup kontainer, termasuk definisi kontainer individual. Definisi grup kontainer memiliki nomor versi. Saat Anda memperbarui definisi grup kontainer, Amazon GameLift Servers menyimpan pembaruan dan menambah nomor versi definisi. Saat mengonfigurasi armada kontainer, Anda dapat menentukan versi definisi grup kontainer mana yang akan diterapkan.

Setelah memperbarui definisi grup kontainer, Anda dapat menerapkan versi baru ke armada kontainer baru atau yang sudah ada.

Perbarui definisi grup kontainer server game

Topik ini menjelaskan cara memperbarui definisi grup kontainer server game menggunakan Amazon GameLift Servers konsol atau alat AWS CLI. Untuk informasi lebih rinci tentang fitur opsional, lihat [Sesuaikan armada Amazon GameLift Servers kontainer](#).

Untuk memperbarui definisi grup kontainer:

Console

Di [Amazon GameLift Servers konsol](#), pilih Wilayah AWS tempat Anda ingin membuat grup kontainer.

Buka bilah navigasi kiri konsol dan pilih Kontainer terkelola: Definisi grup. Pada halaman definisi grup Container, pilih definisi grup kontainer dan versi yang akan diperbarui.

Setelah menyimpan pembaruan, Anda dapat menggunakan versi baru untuk membuat armada kontainer baru atau Anda dapat menerapkan pembaruan ke armada kontainer yang ada.

Langkah 1: Tentukan detail definisi grup kontainer

- Anda dapat memperbarui memori total dan pengaturan batas vCPU.

Langkah 2: Tambahkan definisi wadah

Anda dapat membuat pembaruan definisi kontainer berikut:

- Perbarui definisi wadah yang ada.
 - Tambahkan definisi wadah dukungan baru.
 - Hapus definisi wadah dukungan.
1. Anda dapat memperbarui URI gambar ECR. Pastikan untuk memperbarui setelan versi Server SDK agar sesuai dengan gambar baru.
 2. Anda dapat memperbarui rentang port kontainer internal sesuai kebutuhan. Perubahan yang Anda buat pada pengaturan ini dapat memengaruhi pengaturan port koneksi armada kontainer saat perubahan ini diterapkan ke armada. Untuk detail selengkapnya, lihat [Konfigurasi koneksi jaringan](#).

Langkah 3: Konfigurasikan dependensi

- Anda dapat mengubah dependensi sesuai kebutuhan. Untuk informasi selengkapnya, lihat [Tetapkan dependensi wadah](#).

Langkah 3: Tinjau dan buat

- Tinjau pembaruan definisi grup kontainer Anda. Gunakan Edit untuk membuat perubahan tambahan di bagian mana pun. Setelah selesai, pilih Buat untuk menghasilkan versi baru dari definisi grup kontainer.

Jika permintaan Anda berhasil, konsol akan menampilkan halaman detail untuk sumber definisi grup kontainer baru. Awalnya statusnya adalah `COPYING`, sebagai Amazon GameLift Servers mulai mengambil snapshot dari semua gambar kontainer untuk grup. Ketika fase ini selesai, status definisi grup kontainer berubah menjadi `READY`. Definisi grup kontainer harus dalam `READY` status sebelum Anda dapat membuat armada kontainer dengannya.

AWS CLI

Saat Anda menggunakan AWS CLI untuk membuat atau memperbarui definisi grup kontainer, pertahankan konfigurasi definisi kontainer Anda dalam file terpisah. JSON Anda dapat mereferensikan file dalam perintah CLI Anda. Lihat [Buat JSON file definisi kontainer](#) contoh skema.

Saat memperbarui definisi, Anda hanya perlu menentukan nilai yang ingin Anda perbarui. Amazon GameLift Servers mempertahankan nilai apa pun yang tidak Anda sertakan dalam permintaan pembaruan Anda. Jika Anda mengubah definisi kontainer. Namun, saat mengubah definisi wadah, berikan satu set lengkap.

Untuk memperbarui definisi grup kontainer

Untuk memperbarui definisi grup kontainer baru, gunakan perintah `update-container-group-definition` CLI. Untuk informasi selengkapnya tentang perintah ini, lihat [update-container-group-definition](#) di Referensi Perintah AWS CLI.

Example : grup kontainer server game

Anda dapat menentukan versi definisi grup kontainer saat mengambil, memperbarui, atau menghapus definisi grup kontainer, atau saat membuat atau memperbarui armada kontainer.

Setiap definisi grup kontainer memiliki properti versi. Selain itu, nilai ARN definisi dan menentukan nomor versi.

Contoh ini menggambarkan permintaan untuk perubahan definisi grup kontainer server game. Ini mengasumsikan bahwa Anda telah membuat file JSON dengan definisi wadah untuk grup ini. Contoh ini menggunakan nilai ARN untuk nama definisi, dan menentukan bahwa pemutakhiran adalah versi 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
```

Mengkloning definisi grup kontainer

Anda dapat menggunakan Amazon GameLift Servers konsol untuk mengkloning definisi grup kontainer yang ada.

Untuk mengkloning grup kontainer

1. Di [Amazon GameLift Servers konsol](#), pergi ke panel navigasi kiri dan pilih grup Container.
2. Pada halaman daftar Container groups, pilih grup kontainer yang ada yang ingin Anda kloning. Setelah Anda memilih grup kontainer, tombol Clone aktif.
3. Pilih Klon. Tindakan ini membuka wizard pembuatan grup kontainer dengan pengaturan yang telah diisi sebelumnya.
4. Masukkan nama baru untuk grup kontainer kloning. Grup kontainer di wilayah yang sama harus memiliki nama yang unik.
5. Melangkah melalui grup kontainer dan halaman definisi kontainer, tinjau, dan Buat grup kontainer baru.

Menghapus definisi grup kontainer untuk Amazon GameLift Servers armada kontainer

Anda memiliki beberapa opsi untuk menghapus definisi grup kontainer. Saat Anda menghapus definisi grup kontainer, tindakan ini juga menghapus semua definisi kontainer dalam grup kontainer.

Definisi grup kontainer dapat memiliki beberapa versi. Versi grup kontainer memiliki nama yang sama, tetapi memiliki nomor versi yang berbeda. Definisi grup kontainer ARNs menentukan nama dan versi.

Anda dapat menentukan versi definisi grup kontainer saat mengambil, memperbarui, atau menghapus definisi grup kontainer, atau saat membuat atau memperbarui armada kontainer. Setiap definisi grup kontainer memiliki properti versi. Selain itu, nilai ARN definisi menentukan nomor versi.

Ada beberapa cara untuk menghapus definisi grup kontainer:

- Anda dapat menghapus semua versi definisi tertentu.
- Anda dapat menghapus satu versi tertentu dari definisi tertentu.
- Anda dapat menyimpan beberapa versi terbaru dan menghapus versi lama dari definisi tertentu. Misalnya, Anda dapat menghapus semua versi yang lebih lama dari versi 5.

Anda dapat menghapus versi definisi grup kontainer hanya jika tidak digunakan dalam armada kontainer.

Untuk menghapus definisi grup kontainer

Console

Di [Amazon GameLift Servers konsol](#), pilih Wilayah AWS tempat Anda ingin membuat grup kontainer.

Buka bilah navigasi kiri konsol dan pilih Kontainer terkelola: Definisi grup. Pada halaman definisi grup Container, pilih definisi yang ingin Anda ubah dan pilih Hapus.

Anda diminta untuk memilih jenis penghapusan yang ingin Anda buat dan tentukan pengaturan tambahan tergantung pada jenis hapus.

AWS CLI

- Untuk menghapus definisi grup kontainer, gunakan perintah `delete-container-group-definition` CLI dan berikan nilai untuk jenis penghapusan yang ingin Anda lakukan. Untuk

informasi selengkapnya tentang perintah ini, lihat [delete-container-group-definition](#) di Referensi Perintah AWS CLI.

Contoh ini menggambarkan permintaan untuk menghapus semua versi definisi grup kontainer server game yang lebih lama dari versi 5.

Example

```
aws gamelift delete-container-group-definition \
  --name MyAdventureGameContainerGroup \
  --version-count-to-retain 5 \
```

Amazon GameLift Servers Armada di mana saja

Gunakan Anywhere armada saat Anda ingin memanfaatkan Amazon GameLift Servers fitur dengan sumber daya hosting Anda sendiri. Armada Anywhere biasanya digunakan sebagai lingkungan pengujian untuk pengembangan berulang atau bersama armada terkelola dalam solusi hosting hybrid.

Armada Anywhere terdiri dari sekumpulan sumber daya komputasi (virtual atau fisik) yang Anda suplai dan kelola. Komputasi dapat berada di lokasi geografis mana pun dengan konektivitas, dari laptop lokal hingga pos terdepan jarak jauh. Saat menyiapkan armada Anywhere, Anda menambahkan komputasi ke armada dengan mendaftarkannya melalui Amazon GameLift Servers. Setiap komputasi terdaftar dengan alamat IP (atau nama DNS) sehingga Amazon GameLift Servers dapat menjalin hubungan dengannya.

Anda menyebarkan perangkat lunak server game ke armada Anywhere dengan menginstalnya di setiap komputasi dan meluncurkan proses server game. Setiap proses server game yang diluncurkan membuat koneksi ke Amazon GameLift Servers layanan dan laporan kesiapan untuk menyelenggarakan sesi permainan. Anda dapat menggunakan manajemen konfigurasi dan perkakas penerapan yang ada untuk menangani penerapan awal dan tugas manajemen host. Ada beberapa tugas tambahan yang diperlukan untuk digunakan dengan Amazon GameLift Servers, termasuk:

- Daftar dan deregister menghitung untuk menambah atau menghapusnya dari armada.
- Pertahankan token up-to-date otentikasi pada semua komputasi. Proses server pada komputasi menggunakannya saat menghubungkan ke Amazon GameLift Servers layanan.

 Note

Secara opsional menyebarkan armada Anywhere Anda dengan Amazon GameLift Servers Agen untuk mengotomatiskan tugas-tugas manajemen utama ini. Lihat [Bekerja dengan Amazon GameLift Servers Agen](#).

Lihat topik berikut tentang cara mengatur dan memelihara armada Anywhere:

- [Peta jalan pengembangan untuk hosting dengan Amazon GameLift Servers Anywhere](#)
- [Peta jalan pengembangan untuk hosting hybrid dengan Amazon GameLift Servers](#)
- [Siapkan untuk pengembangan berulang dengan Amazon GameLift Servers Dimanapun](#)
- [Buat sebuah Amazon GameLift Servers Armada di mana saja](#)
- [Cara kerja pembuatan Amazon GameLift Servers armada](#)
- [Perbarui konfigurasi Amazon GameLift Servers armada](#)

Alur kerja pembuatan armada di mana saja

Untuk armada di mana saja, Amazon GameLift Servers mengatur sumber daya armada saja. Anda mengatur dan mendaftarkan komputasi dengan armada, dan Anda menginstal perangkat lunak server game dan memulai proses server game untuk meng-host sesi game.

1. Amazon GameLift Servers menciptakan sumber daya armada di wilayah asal armada. Status armada dan status lokasi kustom diatur ke Baru.
2. Amazon GameLift Servers mulai menulis acara ke log peristiwa armada.
3. Setelah sumber daya armada dibuat, Amazon GameLift Servers menetapkan status armada ke Aktif. Pada titik ini, Anda dapat mendaftarkan komputasi baru dengan armada.

Buat sebuah Amazon GameLift Servers Armada di mana saja

Topik ini menjelaskan cara membuat Amazon GameLift Servers Armada di mana saja. Dengan armada Anywhere, Anda dapat menggunakan inti Amazon GameLift Servers fitur manajemen sesi game saat menyelenggarakan sesi game dengan sumber daya komputasi Anda sendiri. Buat armada Anywhere untuk perangkat keras lokal atau sumber daya berbasis Internet lainnya.

Armada di mana saja biasanya digunakan bersama Amazon GameLift Servers armada terkelola dalam solusi hosting hybrid. Mereka juga menyediakan lingkungan pengujian yang berguna saat mengembangkan game untuk hosting Amazon GameLift Servers. Lihat topik ini untuk mempelajari lebih lanjut tentang kapan dan bagaimana menggabungkan Amazon GameLift Servers Di mana saja armada menjadi solusi hosting game:

- [Hosting hibrida](#)
- [Menyiapkan armada hosting dengan Amazon GameLift Servers](#)
- [Siapkan untuk pengembangan berulang dengan Amazon GameLift Servers Dimanapun](#)

Karena armada Anywhere dikelola sendiri, menyiapkan armada membutuhkan beberapa pekerjaan tambahan. Untuk mendapatkan armada Anywhere yang siap menjadi tuan rumah sesi permainan dan pemain, Anda harus menyelesaikan tugas-tugas berikut:

Topik

- [Sebelum Anda mulai](#)
- [Buat lokasi kustom](#)
- [Buat armada Anywhere](#)
- [Tambahkan komputasi ke armada](#)
- [Memulai server game](#)

Sebelum Anda mulai

Sebelum membuat armada Anywhere, lakukan tugas-tugas berikut. Untuk panduan lebih rinci, lihat [Peta jalan pengembangan untuk hosting dengan Amazon GameLift Servers Anywhere](#) atau [Peta jalan pengembangan untuk hosting hybrid dengan Amazon GameLift Servers](#).

- Integrasikan kode server game Anda dengan Amazon GameLift Servers server SDK versi 5.x (atau lebih tinggi). Anda tidak perlu menyelesaikan semua tugas integrasi game, hanya yang diperlukan untuk membangun server game. Praktik umum adalah mengatur mesin lokal Anda sebagai armada Anywhere dan menggunakan antarmuka baris perintah untuk menguji integrasi server game Anda (lihat [Siapkan pengujian lokal dengan Amazon GameLift Servers Anywhere](#)). Anda dapat menggabungkan komponen tambahan (seperti Amazon GameLift Servers mengaktifkan klien game) saat Anda mengembangkannya.

- Package perangkat lunak server game Anda untuk instalasi ke komputasi armada Anywhere Anda. Paket harus menyertakan build server game terintegrasi Anda dan semua perangkat lunak pendukung yang diperlukan untuk menjalankan server game Anda.
- Putuskan apakah akan menggunakan Amazon GameLift Servers Agen dengan armada Anywhere Anda. Agen adalah alat manajemen proses on-compute yang mengotomatiskan beberapa tugas utama yang terkait dengan mengelola proses server dan komputasi untuk digunakan dengan Amazon GameLift Servers. Untuk informasi selengkapnya, lihat [Bekerja dengan Amazon GameLift Servers Agen](#).

Buat lokasi kustom

Buat lokasi khusus untuk mewakili lokasi fisik sumber daya komputasi Anda. Saat membuat armada Anywhere, Anda harus memiliki setidaknya satu lokasi kustom yang sudah ditentukan. Anda dapat membuat lokasi kustom tambahan dan menambahkannya ke armada yang ada kapan saja.

Untuk membuat lokasi kustom

Gunakan salah satu Amazon GameLift Servers konsol atau AWS Command Line Interface (AWS CLI) untuk membuat lokasi kustom.

Console

Di [Amazon GameLift Servers konsol](#), gunakan panel navigasi untuk membuka halaman Lokasi. Pilih Buat lokasi untuk membuka kotak dialog Buat.

1. Di kotak dialog, masukkan nama Lokasi. Sebagai praktik terbaik, gunakan nama yang menggambarkan lokasi yang bermakna untuk sekumpulan sumber daya komputasi. Mungkin lokasi geografis, nama pusat data, atau pengenalan lokasi lainnya. Amazon GameLift Servers menambahkan nama lokasi kustom Anda dengan custom-.
2. (Opsional) Tambahkan tag ke lokasi kustom Anda. Setiap tag terdiri dari kunci dan nilai opsional, yang keduanya Anda tentukan. Tetapkan tag ke AWS sumber daya yang ingin Anda kategorikan dengan cara yang berguna, seperti berdasarkan tujuan, pemilik, atau lingkungan. Pilih Tambahkan tag baru untuk setiap tag yang ingin Anda tambahkan.
3. Pilih Buat.

AWS CLI

Buat lokasi khusus menggunakan [create-location](#) perintah. Berikan `location-name` nilai, yang harus dimulai dengan `custom-`. Sebagai praktik terbaik, gunakan nama yang menggambarkan lokasi yang bermakna untuk sekumpulan sumber daya komputasi. Mungkin lokasi geografis, nama pusat data, atau pengenalan lokasi lainnya.

```
aws gamelift create-location \
  --location-name custom-location-1
```

Output

```
{
  "Location": {
    "LocationName": "custom-location-1",
    "LocationArn": "arn:aws:gamelift:us-east-1:111122223333:location/custom-
location-1"
  }
}
```

Buat armada Anywhere

Buat armada Anywhere untuk satu set sumber daya komputasi yang Anda miliki. Armada Anywhere baru mulai kosong; Anda menambahkan komputasi ke armada dengan mendaftarkannya.

Saat pembuatan, armada Anywhere baru dengan cepat bergerak melalui status armada dari `NEW` ke `ACTIVE`. Anda dapat menambahkan komputasi ke armada setelah mencapai `ACTIVE`.

Untuk membuat armada Anywhere

Gunakan salah satu Amazon GameLift Servers konsol atau AWS Command Line Interface (AWS CLI) untuk membuat armada Anywhere.

Console

Di [Amazon GameLift Servers konsol](#), gunakan panel navigasi untuk membuka halaman Armada. Pilih Buat armada untuk memulai alur kerja pembuatan armada.

Langkah 1 Pilih jenis komputasi

Pilih opsi Anywhere dan pilih Next.

Langkah 2 Tentukan detail armada

Pada langkah ini, tentukan beberapa pengaturan utama di seluruh armada.

1. Isi bagian Detail Armada:
 - a. Masukkan Nama armada. Sebaiknya gunakan pola penamaan armada yang memudahkan identifikasi jenis armada saat melihat daftar armada.
 - b. Berikan Deskripsi singkat tentang armada.
2. Tetapkan opsional ini Detail tambahan sesuai kebutuhan. Anda dapat memperbarui pengaturan armada ini nanti.
 - a. Saat membuat armada untuk pengujian produksi atau pra-prod, gunakan pengaturan ini untuk menentukan nilai Biaya per jam untuk perhitungan armada. Amazon GameLift Servers dapat menggunakan informasi ini selama proses penempatan sesi permainan untuk memilih sumber daya hosting berdasarkan biaya.
 - b. Jika Anda ingin menggabungkan data metrik untuk armada ini dan lainnya, tentukan nama grup Metrik. Gunakan nama grup metrik yang sama untuk semua armada yang ingin Anda gabungkan bersama. Lihat metrik untuk grup metrik untuk melihat data agregat.
3. Tambahkan tag opsional ke lokasi kustom Anda. Setiap tag terdiri dari kunci dan nilai opsional, yang keduanya Anda tentukan. Tetapkan tag ke AWS sumber daya yang ingin Anda kategorikan dengan cara yang berguna, seperti berdasarkan tujuan, pemilik, atau lingkungan. Pilih Tambahkan tag baru untuk setiap tag yang ingin Anda tambahkan.
4. Pilih Berikutnya untuk melanjutkan alur kerja.

Langkah 3 Pilih lokasi kustom

Pada langkah ini, identifikasi lokasi fisik komputasi yang Anda rencanakan untuk ditambahkan ke armada ini. Anda dapat menentukan satu atau beberapa lokasi sekarang, dan Anda dapat menambahkan atau menghapus lokasi nanti sesuai kebutuhan.

1. Di lokasi Kustom, pilih satu atau beberapa lokasi untuk komputasi armada. Daftar ini mencakup semua lokasi kustom yang telah ditentukan dalam pilihan Anda saat ini Wilayah AWS. Untuk menentukan lokasi kustom baru yang ingin ditambahkan ke armada, pilih Buat lokasi.
2. Pilih Berikutnya untuk melanjutkan alur kerja.

Langkah 4 Tinjau dan buat

Tinjau pengaturan Anda sebelum membuat armada.

Saat Anda siap untuk menyebarkan armada baru, pilih Buat. Amazon GameLift Servers segera memulai proses aktivasi armada, menetapkan ID unik dan menempatkan armada dalam NEW status. Anda dapat melacak kemajuan armada di halaman Armada.

AWS CLI

Gunakan [create-fleet](#) perintah untuk membuat armada tipe ANYWHERE komputasi. Berikan nama dan setidaknya satu lokasi khusus. Amazon GameLift Servers membuat sumber daya armada Anywhere di default Anda saat ini Wilayah AWS (atau Anda dapat menambahkan tag --region untuk menentukan yang berbeda Wilayah AWS).

Contoh permintaan berikut membuat armada baru dengan pengaturan minimal yang diperlukan. Ganti *FleetName* dan *custom-location* dengan informasi Anda sendiri.

```
aws gamelift create-fleet \  
--name FleetName \  
--compute-type ANYWHERE \  
--locations "Location=custom-location"
```

Contoh tanggapan

```
{  
  "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"  
  }  
}
```



```
}
```

Saat pembuatan, armada Anywhere baru dengan cepat bergerak ke status armadaACTIVE. Anda dapat menambahkan komputasi ke armada setelah mencapaiACTIVE.

Perhatikan bahwa respon tidak termasuk lokasi armada. Anda dapat mengambil detail armada lengkap dengan menelepon [describe-fleet-attributes](#) dan [describe-fleet-location-attributes](#).

Tambahkan komputasi ke armada

Untuk menambahkan sumber daya komputasi ke armada dan menyiapkannya untuk menyelenggarakan sesi permainan, lakukan tugas-tugas berikut:

- Daftarkan komputasi dengan armada. Registrasi memberitahu Amazon GameLift Servers sumber daya hosting fisik apa yang merupakan bagian dari armada.
- Minta token otentikasi untuk komputasi. Setiap server game yang berjalan pada komputasi membutuhkan token ini untuk terhubung ke Amazon GameLift Servers layanan. Token otentikasi bersifat sementara dan harus diperbarui secara teratur.

Note

Jika Anda menggunakan perangkat lunak server game Anda dengan Amazon GameLift Servers Agen, Anda dapat melewati langkah ini. Agen secara otomatis mendaftarkan setiap komputasi dan mempertahankan token otentikasi yang valid untuk komputasi. Lihat [Bekerja dengan Amazon GameLift Servers Agen](#).

Anda dapat mendaftarkan komputasi dan meminta token otentikasi dengan menggunakan AWS CLI atau membuat panggilan terprogram ke SDK untuk AWS Amazon GameLift Servers. Tindakan ini tidak tersedia melalui Amazon GameLift Servers konsol.

Sebagai praktik terbaik, kami sarankan untuk mengotomatiskan kedua tugas ini dengan menambahkan skrip startup ke setiap komputasi. Skrip startup secara otomatis memanggil perintah `register-compute` dan `get-compute-auth-token` perintah. Anda juga dapat mengotomatiskan tugas untuk menyegarkan token autentikasi secara teratur sepanjang masa komputasi dan membatalkan pendaftaran komputasi saat dimatikan.

Setiap tindakan startup mengembalikan nilai khusus komputasi yang perlu Anda simpan di komputasi. Ketika proses server game diluncurkan pada komputasi, itu harus meneruskan nilai-nilai ini sebagai parameter server saat menginisialisasi koneksi dengan Amazon GameLift Servers layanan (lihat [ServerParameters](#) di referensi SDK server). Kami menyarankan Anda menetapkan nilai khusus komputasi ini (atau lokasi yang disimpan) sebagai variabel lingkungan. Jika Anda menggunakan Amazon GameLift Servers Agen, tugas ini ditangani untukmu. Nilai khusus komputasi adalah sebagai berikut:

- `register-compute` mengembalikan nilai untuk `GameLiftServiceSdkEndpoint`. Tetapkan nilai ini ke parameter `WebSocketUrl` server.
- `compute-auth-token` mengembalikan token otentikasi. Tetapkan nilai ini ke parameter `authToken` server.

AWS CLI

Petunjuk berikut menjelaskan cara mengirimkan setiap permintaan secara manual menggunakan AWS CLI.

Untuk mendaftarkan komputasi

Hubungi [register-compute](#) untuk mendaftarkan komputasi. Identifikasi ID armada untuk menambahkan komputasi. Berikan informasi komputasi berikut: nama yang bermakna, alamat IP, dan lokasi. Lokasi komputasi harus merupakan lokasi khusus yang sudah dikaitkan dengan armada. Jika Anda ingin menggunakan lokasi kustom yang berbeda, gunakan Amazon GameLift Servers konsol untuk memperbarui armada atau memanggil perintah AWS CLI [create-fleet-locations](#) untuk menambahkan lokasi khusus ke armada.

Dalam contoh berikut, ganti nilai placeholder untuk komputasi dan armada Anda. `fleet-id` Nilai dikembalikan saat Anda membuat armada Anywhere. Anda dapat mengambil detail armada lengkap dengan menelepon [describe-fleet-attributes](#) dan [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
```

Contoh Output

```
{
  "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"
  }
}
```

Untuk meminta token otentikasi

Panggilan [get-compute-auth-token](#) untuk meminta token otentikasi yang valid. daftarkan komputasi. Identifikasi ID armada dan hitung nama.

Dalam contoh berikut, ganti nilai placeholder untuk komputasi dan armada Anda. `fleet-id` Nilai dikembalikan saat Anda membuat armada Anywhere. Anda dapat mengambil detail armada lengkap dengan menelepon [describe-fleet-attributes](#). Untuk menemukan informasi komputasi, hubungi [list-compute](#) dengan ID armada untuk melihat semua komputasi yang terdaftar ke armada.

```
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
```

Contoh Output

```
{
  "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"
}
```

Memulai server game

Setelah Anda membuat armada Anywhere dan menambahkan satu atau lebih komputasi ke armada, Anda siap untuk mulai menjalankan server game Anda.

Langkah 1 Instal perangkat lunak server game Anda

Dapatkan build server game Anda dan semua perangkat lunak dependen diinstal ke setiap komputasi di armada Anywhere Anda. Pembuatan server game harus terintegrasi dengan Amazon GameLift Servers server SDK versi 5.x (atau lebih tinggi) dengan fungsionalitas minimum yang diperlukan untuk berkomunikasi dengan Amazon GameLift Servers layanan.

Langkah 2 Siapkan komputasi Anda untuk menjalankan server game

Pastikan bahwa setiap komputasi terdaftar dan memiliki token otentikasi yang valid. Jika Anda menggunakan skrip untuk mengelola tugas ini, pastikan skrip berjalan di setiap komputasi sebelum memulai proses server game apa pun.

Jika Anda telah menerapkan Amazon GameLift Servers Agen dengan perangkat lunak server game Anda, pastikan Agen yang dapat dieksekusi diluncurkan.

Langkah 3 Luncurkan proses server game

Jalankan instance server game Anda yang dapat dieksekusi pada komputasi. Jika build server game Anda terintegrasi dengan benar, proses server game akan memanggil tindakan SDK server `InitSDK()` dengan serangkaian parameter server yang valid. Ketika proses server siap untuk meng-host sesi permainan, ia memanggil `ProcessReady()`.

Note

Jika Anda menggunakan perangkat lunak server game Anda dengan Amazon GameLift Servers Agen, Anda dapat melewati langkah ini. Agen secara otomatis meluncurkan proses server game berdasarkan instruksi runtime yang Anda berikan.

Anda dapat memantau kemajuan dengan melihat metrik proses server untuk mengaktifkan dan mengaktifkan proses server. Lihat [Amazon GameLift Serversmetrik untuk armada](#). Jika proses server game Anda gagal diinisialisasi, verifikasi bahwa proses tersebut mengambil nilai parameter server yang tepat untuk komputasi yang dijalankannya.

Abstrak sebuah Amazon GameLift Servers penunjukan armada dengan alias

Sesi Amazon GameLift Servers alias digunakan untuk mengabstraksi tujuan hosting. Tujuan hosting memberi tahu Amazon GameLift Servers tempat mencari sumber daya yang tersedia untuk menyelenggarakan sesi permainan baru untuk pemain. Alias berguna dalam skenario berikut:

- Jika game Anda tidak menggunakan antrian multi-armada untuk penempatan sesi game, ia meminta sesi permainan baru dengan menentukan Amazon GameLift Servers ID armada. Selama masa permainan, Anda akan mengganti armada beberapa kali, untuk memperbarui build server, memperbarui perangkat keras dan sistem operasi hosting, atau menyelesaikan masalah kinerja. Gunakan alias untuk mengabstraksi ID armada sehingga Anda dapat dengan mulus mengalihkan lalu lintas pemain dari armada yang ada ke armada baru.
- Jika Anda ingin melakukan sesuatu selain membuat sesi permainan baru saat klien game memintanya. Misalnya, Anda mungkin ingin mengarahkan pemain yang menggunakan out-of-date klien ke situs web peningkatan.

Alias harus menentukan strategi routing. Ada dua jenis. Strategi perutean sederhana merutekan lalu lintas pemain ke ID armada tertentu, yang dapat Anda perbarui untuk mengarahkan lalu lintas. Strategi perutean terminal meneruskan pesan kembali ke klien alih-alih membuat sesi permainan baru. Anda dapat mengubah strategi routing alias kapan saja.

Jika Anda menggunakan antrian untuk penempatan sesi game, Anda tidak memerlukan alias untuk mengarahkan lalu lintas saat mengganti armada. Dengan antrian, Anda cukup menambahkan armada baru dan menghapus armada lama. Tindakan ini tidak terlihat oleh pemain, karena permintaan sesi permainan baru secara otomatis dipenuhi menggunakan armada baru. Itu tidak memengaruhi sesi permainan yang ada. Anda dapat mengidentifikasi tujuan antrian dengan menggunakan ID armada atau alias.

Topik

- [Buat sebuah Amazon GameLift Servers alias](#)
- [Mengedit alias](#)

Buat sebuah Amazon GameLift Servers alias

Topik ini menjelaskan cara membuat Amazon GameLift Servers alias untuk digunakan dengan penempatan sesi game.

Untuk membuat alias

Gunakan salah satu Amazon GameLift Servers konsol atau AWS Command Line Interface (AWS CLI) untuk membuat alias.

Console

Di [Amazon GameLift Servers konsol](#), gunakan panel navigasi untuk membuka halaman Alias.

1. Pilih Buat alias.
2. Masukkan nama alias. Sebaiknya sertakan karakteristik yang berarti dalam nama alias untuk membantu saat melihat daftar alias.
3. Masukkan alias Deskripsi sesuai kebutuhan.
4. Pilih strategi Routing untuk alias.
 - a. Jika Anda memilih strategi perutean sederhana, pilih ID armada dari daftar untuk dikaitkan dengan alias ini. Daftar ini mencakup semua armada dalam eith yang saat ini dipilih. Wilayah AWS Anda harus membuat alias di Wilayah yang sama dengan armada.
 - b. Jika Anda memilih strategi perutean Terminal, masukkan nilai string yang Anda inginkan Amazon GameLift Servers untuk kembali ke klien game sebagai tanggapan atas permintaan sesi permainan. Permintaan dengan alias terminal melempar pengecualian dengan pesan yang disematkan.
5. (Opsional) Tambahkan Tag ke sumber daya alias. Setiap tag terdiri dari kunci dan nilai opsional, yang keduanya Anda tentukan. Tetapkan tag ke AWS sumber daya yang ingin Anda kategorikan dengan cara yang berguna, seperti berdasarkan tujuan, pemilik, atau lingkungan. Pilih Tambahkan tag baru untuk setiap tag yang ingin Anda tambahkan.
6. Saat Anda siap untuk menyebarkan armada baru, pilih Buat.

AWS CLI

Gunakan [create-alias](#) perintah untuk membuat alias. Amazon GameLift Servers membuat sumber daya alias di default Anda saat ini Wilayah AWS (atau Anda dapat menambahkan tag --region untuk menentukan yang berbeda Wilayah AWS).

Minimal, sertakan nama alias dan strategi perutean. Untuk strategi routing sederhana, tentukan ID armada di Region yang sama dengan alias. Untuk strategi routing terminal, berikan string pesan.

Mengedit alias

Anda dapat mengedit alias menggunakan Amazon GameLift Servers [konsol atau dengan perintah AWS CLI update-alias](#).

Topik ini menjelaskan cara mengedit Amazon GameLift Servers alias untuk digunakan dengan penempatan sesi game. Anda dapat melakukan pengeditan berikut:

Untuk mengedit alias

Gunakan salah satu Amazon GameLift Servers konsol atau AWS Command Line Interface (AWS CLI) untuk mengedit alias.

Console

Di [Amazon GameLift Servers konsol](#), gunakan panel navigasi untuk membuka halaman Alias.

1. Pilih alias yang ingin Anda edit dan pilih Edit. Jika Anda tidak melihat alias yang ingin Anda edit, periksa yang Anda pilih Wilayah AWS saat ini.
2. Pada halaman Edit alias, Anda dapat melakukan pengeditan berikut:
 - Ubah nama alias.
 - Ubah deskripsi alias.
 - Ubah strategi routing dari simple ke terminal, atau dari terminal ke simple.
 - Untuk alias dengan strategi routing sederhana, ubah ID armada yang terkait dengan alias.
 - Untuk alias dengan strategi routing terminal, ubah teks pesan.
3. Pilih Simpan perubahan. Saat memperbarui ID armada untuk alias dengan strategi perutean sederhana, mungkin diperlukan waktu hingga 2 menit untuk menyelesaikan transisi. Selama waktu ini, penempatan sesi permainan baru mungkin terjadi di armada lama.

AWS CLI

Gunakan [update-alias](#) perintah untuk membuat perubahan pada sumber daya alias. Anda dapat memperbarui sumber daya alias di default Anda saat ini Wilayah AWS, atau Anda dapat menambahkan `--region` tag untuk menentukan yang berbeda Wilayah AWS.

Anda dapat mengubah properti berikut:

- Nama alias.
- Deskripsi alias.
- Jenis strategi routing. Pastikan untuk memberikan ID armada atau string pesan untuk strategi routing baru.
- Fleet ID untuk strategi routing sederhana yang ada. ID armada harus berada di wilayah yang sama dengan alias.
- String pesan untuk strategi perutean terminal yang ada.

Mengelola penempatan sesi permainan dengan Amazon GameLift Servers mengantre

Antrian sesi permainan adalah mekanisme utama yang Amazon GameLift Servers digunakan untuk mencari server game yang tersedia dan memilihnya untuk menjadi tuan rumah sesi permainan baru. Antrian menawarkan cara yang jauh lebih efisien untuk memproses sejumlah besar permintaan sesi game dan menemukan penempatan untuk mereka di berbagai armada sumber daya hosting. Jika solusi hosting Anda menggunakan lebih dari satu armada, dan Anda memproses permintaan dengan volume tinggi, Anda mungkin memerlukan antrian.

Ketika game Anda ingin memulai sesi permainan baru untuk pemain, itu mengirimkan permintaan penempatan ke Amazon GameLift Servers layanan, yang menyalurkannya ke antrian. Konfigurasi antrian menentukan kapan dan bagaimana permintaan diproses. Saat memproses permintaan penempatan, Amazon GameLift Servers mencari satu set armada untuk server game untuk menjadi tuan rumah sesi permainan. Penempatan berhasil ketika Amazon GameLift Servers menemukan server game yang tersedia dan memintanya untuk memulai sesi permainan.

Topik

- [Karakteristik antrian](#)
- [Buat antrean sesi game](#)
- [Kustomisasi antrian sesi permainan](#)
- [Atur notifikasi kejadian untuk penempatan sesi game](#)
- [Tutorial: Membuat Amazon GameLift Servers antrian dengan Instans Spot](#)

Karakteristik antrian

Sesi Amazon GameLift Servers antrian sesi permainan adalah sumber daya AWS cloud. Anda dapat membuat antrian di semua Wilayah AWS itu Amazon GameLift Servers mendukung (lihat [Amazon GameLift Servers lokasi layanan](#)). Permintaan penempatan sesi permainan dikirim ke lokasi tersebut dan diproses di sana.

Mengotomatiskan penempatan sesi game dengan antrian menawarkan manfaat signifikan bagi pengembang dan pemain game. Ini termasuk:

- Antrian memberikan penempatan “terbaik”. Saat memproses permintaan penempatan sesi game, antrian menggunakan Amazon GameLift Servers Algoritma FleeTiQ untuk memprioritaskan penempatan berdasarkan serangkaian preferensi yang ditentukan, termasuk biaya, lokasi, dan latensi pemain.
- Antrian mendukung armada Spot untuk membantu mengurangi biaya hosting game. Anda dapat mengonfigurasi antrian Anda dengan armada AWS Spot, yang sering menawarkan biaya hosting yang jauh lebih rendah, serta armada On-Demand. Karena biaya rendah adalah salah satu kriteria utama untuk penempatan, antrian selalu dapat memanfaatkan perbedaan biaya.
- Antrian dapat menempatkan game baru lebih cepat selama permintaan tinggi. Dengan mengonfigurasi antrian dengan beberapa armada, Anda menyediakan opsi yang lebih fleksibel untuk penempatan sesi game. Tetapi armada tambahan juga menyediakan kapasitas cadangan sesuai kebutuhan saat permintaan meningkat. Untuk setiap permintaan penempatan, jika Amazon GameLift Servers tidak dapat menempatkan sesi permainan di lokasi yang paling disukai, secara otomatis bergerak untuk mengevaluasi lokasi lain.
- Antrian dapat membuat ketersediaan server game lebih tangguh. Pemadaman bisa terjadi. Dengan antrian multi-armada, perlambatan atau pemadaman tidak harus memengaruhi akses pemain ke game Anda. Dengan mengonfigurasi antrian Anda dengan armada yang memiliki kapasitas di zona ketersediaan Wilayah AWS dan berbeda, Anda dapat membantu memastikan bahwa pemain selalu dapat menemukan sesi permainan untuk bergabung.
- Dapatkan metrik tentang penempatan sesi game dan kinerja antrian. Amazon GameLift Servers memancarkan metrik antrian, termasuk statistik tentang keberhasilan dan kegagalan penempatan, jumlah permintaan dalam antrian, dan waktu rata-rata yang dihabiskan permintaan dalam antrian. Anda dapat melihat metrik ini di Amazon GameLift Servers konsol atau di CloudWatch.

Untuk memulai dengan membuat antrian pemula dasar, lihat [Buat antrean sesi game](#).

Buat antrean sesi game

Antrian digunakan untuk menempatkan sesi permainan baru di beberapa armada dan lokasi. Game Anda memulai sesi permainan baru dengan mengirimkan permintaan penempatan ke antrian. Antrian dikonfigurasi dengan instruksi untuk cara memproses permintaan. Pelajari lebih lanjut tentang memulai permintaan penempatan sesi game di [Buat sesi permainan](#).

Untuk membuat antrian sesi permainan

Instruksi ini menggambarkan cara membuat antrian kerja sederhana dengan pengaturan konfigurasi minimal dan pengaturan default. Ada sejumlah opsi untuk menyesuaikan konfigurasi antrian. Opsi ini membantu Anda membuat penempatan terbaik berdasarkan kebutuhan permainan Anda. Untuk mempelajari lebih lanjut tentang menyesuaikan antrian untuk game Anda, lihat. [Kustomisasi antrian sesi permainan](#) Anda dapat memperbarui sebagian besar pengaturan konfigurasi antrian kapan saja.

Anda dapat membuat antrian sesi game menggunakan Amazon GameLift Servers konsol atau AWS CLI.

Console

Di [Amazon GameLift Servers konsol](#), pilih AWS Wilayah untuk bekerja. Buka bilah navigasi kiri konsol dan pilih Antrian.

1. Pada halaman Antrian, pilih Buat antrian untuk memulai alur kerja.
2. Di bawah Pengaturan antrian masukkan pengaturan berikut:
 - a. Masukkan nama antrian. Nama ini harus unik untuk tempat Wilayah AWS Anda membuat antrean.
 - b. Pertahankan pengaturan Timeout default, yaitu 600 detik (atau 10 menit). Nilai ini mengontrol berapa lama Amazon GameLift Servers mencoba untuk menempatkan sesi permainan baru sebelum berhenti. Amazon GameLift Servers mencari sumber daya yang tersedia hingga waktu permintaan habis. Anda dapat memperbarui pengaturan batas waktu antrian kapan saja.
 - c. Lewati bagian Kebijakan latensi pemain. Antrian menggunakan kebijakan latensi hanya jika menerima permintaan penempatan yang menyertakan data latensi pemain. Anda dapat menambahkan kebijakan latensi ke antrian kapan saja. Untuk selengkapnya tentang membuat kebijakan latensi, lihat [Buat kebijakan latensi pemain](#).
3. Lewati bagian Lokasi penempatan sesi permainan untuk menggunakan pengaturan default Semua lokasi. Pengaturan ini memungkinkan Anda membuat daftar lokasi yang memungkinkan antrean dapat membuat penempatan (juga disebut konfigurasi filter). Untuk selengkapnya tentang memprioritaskan berdasarkan lokasi dan konfigurasi filter, lihat. [Prioritaskan penempatan berdasarkan lokasi](#)
4. Di bawah Urutan tujuan, tambahkan satu atau beberapa armada ke antrian. Anda dapat mengidentifikasi armada dengan menggunakan armada IDs atau ARNs, atau dengan menggunakan alias armada. Saat menambahkan beberapa armada, perlu diingat bahwa mereka semua harus menjalankan build game yang serupa dan kompatibel dengan klien

game apa pun yang menggunakan antrian ini. Selain itu, semua armada dalam antrian harus memiliki konfigurasi sertifikat yang sama.

- a. Pilih Wilayah tempat armada atau alias dibuat. Untuk armada multi-lokasi, ini adalah wilayah “rumah”.
 - b. Untuk Jenis tujuan, pilih armada atau alias.
 - c. Pilihan wilayah dan tipe Anda mengisi daftar dropdown armada atau alias yang ada. Pilih satu untuk ditetapkan sebagai tujuan antrian.
 - d. Untuk menentukan armada atau alias lain untuk antrian, pilih Tambah tujuan dan ulangi langkah sebelumnya.
 - e. Setelah menambahkan daftar tujuan, gunakan drag-and-drop fitur tersebut untuk menyusun ulang tujuan. Amazon GameLift Servers menggunakan pesanan ini saat memprioritaskan penempatan berdasarkan tujuan.
5. Lewati bagian Prioritas penempatan sesi Game untuk menjaga urutan prioritas default. Pengaturan ini memungkinkan Anda menyesuaikan cara Amazon GameLift Servers memilih tempat mencari sumber daya hosting yang tersedia untuk penempatan sesi game baru. Untuk informasi selengkapnya tentang memprioritaskan penempatan, lihat [Prioritaskan penempatan sesi permainan](#) Anda dapat memperbarui prioritas penempatan antrian kapan saja.
 6. Di bawah Urutan lokasi, pertahankan nilai default. Pengaturan ini digunakan saat memprioritaskan berdasarkan lokasi armada. Ini menyediakan urutan lokasi untuk digunakan. Saat menggunakan pengaturan prioritas default, lokasi digunakan sebagai tiebreaker ketika tujuan pilihan adalah armada dengan beberapa lokasi.
 7. Lewati bagian Pengaturan pemberitahuan Acara opsional. Pemberitahuan acara diperlukan untuk antrian yang memproses permintaan penempatan volume tinggi. Untuk antrian yang memproses volume rendah, seperti untuk tujuan pengembangan atau pengujian, Anda dapat melacak status permintaan penempatan dengan melakukan polling. [DescribeGameSessionPlacement](#) Untuk detail selengkapnya, lihat [Atur notifikasi kejadian untuk penempatan sesi game](#). Anda dapat memperbarui pengaturan pemberitahuan acara antrian kapan saja.
 8. Pilih Buat untuk menghasilkan antrian baru dengan kustomisasi minimal.

AWS CLI

Example Membuat antrean

Contoh berikut membuat antrian sesi permainan dengan konfigurasi ini:

- Batas waktu lima menit.
- Dua tujuan armada.
- Filter untuk hanya mengizinkan penempatan di lokasi ini: `us-east-1`, `us-east-2`, `us-west-2`, dan `ca-central-1`.
- Urutan prioritas berdasarkan biaya dan kemudian lokasi dalam urutan tertentu.

```
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

Anda bisa mendapatkan armada dan alias nilai ARN dengan memanggil [describe-fleet-attributes](#) salah satu atau deskripsi-alias dengan armada [atau](#) alias ID.

Jika `create-game-session-queue` permintaan berhasil, Amazon GameLift Servers mengembalikan [GameSessionQueue](#) objek dengan konfigurasi antrian baru. Anda sekarang dapat mengirimkan permintaan ke antrian menggunakan [StartGameSessionPlacement](#).

Example Buat antrian dengan kebijakan latensi pemain

Contoh berikut membuat antrian sesi permainan dengan konfigurasi ini:

- Batas waktu sepuluh menit

- Tiga tujuan armada
- Satu set kebijakan latensi pemain

```
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" \  

```

Jika `create-game-session-queue` permintaan berhasil, Amazon GameLift Servers mengembalikan [GameSessionQueue](#) objek dengan konfigurasi antrian baru.

Kustomisasi antrian sesi permainan

Topik ini menjelaskan cara menyesuaikan antrian sesi permainan Anda untuk membuat keputusan terbaik tentang penempatan sesi game. Untuk informasi lebih lanjut tentang antrian sesi permainan dan cara kerjanya, lihat. [Mengelola penempatan sesi permainan dengan Amazon GameLift Servers mengantre](#)

Amazon GameLift Servers Fitur-fitur ini membutuhkan antrian:

- [Perjodohan dengan FlexMatch](#)
- [Desain antrian untuk Instans Spot](#)

Topik

- [Praktik terbaik untuk Amazon GameLift Servers antrian sesi permainan](#)
- [Tentukan ruang lingkup antrian](#)
- [Membangun antrian multi-lokasi](#)

- [Prioritaskan penempatan sesi permainan](#)
- [Evaluasi metrik antrean](#)
- [Desain antrian untuk Instans Spot](#)

Praktik terbaik untuk Amazon GameLift Servers antrian sesi permainan

Antrian sesi permainan berisi daftar armada tempat Amazon GameLift Servers dapat menempatkan sesi permainan baru. Setiap armada dapat memiliki sumber daya hosting yang digunakan di beberapa lokasi geografis. Saat memilih penempatan, antrian memilih armada dan lokasi armada berdasarkan serangkaian prioritas yang Anda tetapkan untuk armada.

Pertimbangkan panduan dan praktik terbaik berikut ini:

- Tambahkan armada di lokasi yang mencakup pemain Anda. Anda dapat menambahkan armada dan alias di lokasi mana pun yang tersedia. Lokasi penting jika Anda membuat penempatan berdasarkan latensi pemain yang dilaporkan.
- Gunakan alias untuk semua armada. Tetapkan alias ke setiap armada dalam antrean, dan gunakan nama alias saat menetapkan tujuan dalam antrean Anda.
- Gunakan build atau skrip game yang sama atau serupa untuk semua armada. Antrean dapat menempatkan pemain ke sesi game pada armada apa pun dalam antrean. Pemain harus dapat bermain dalam sesi game apa pun di armada mana pun.
- Buat armada di setidaknya dua lokasi. Dengan memiliki server game yang dihosting di setidaknya satu lokasi lain, Anda mengurangi dampak pemadaman Regional pada pemain Anda. Anda dapat menjaga armada cadangan Anda diperkecil, dan menggunakan penskalaan otomatis untuk meningkatkan kapasitas jika penggunaan meningkat.
- Prioritaskan penempatan sesi permainan Anda. Sebuah antrean memprioritaskan pilihan penempatan berdasarkan beberapa elemen, termasuk urutan daftar tujuan.
- Buat antrian Anda di lokasi yang sama dengan layanan klien Anda. Dengan menempatkan antrian Anda di lokasi dekat layanan klien Anda, Anda dapat meminimalkan latensi komunikasi.
- Gunakan armada dengan beberapa lokasi. Gunakan konfigurasi filter antrean untuk mencegah antrean menempatkan sesi game di lokasi tertentu. Anda dapat menggunakan setidaknya dua armada multi-lokasi dengan lokasi rumah yang berbeda untuk mengurangi dampak penempatan game selama pemadaman Regional.
- Gunakan pengaturan sertifikat TLS yang sama untuk semua armada. Klien game yang terhubung ke sesi permainan di armada Anda harus memiliki protokol komunikasi yang kompatibel.

Tentukan ruang lingkup antrian

Populasi pemain game Anda mungkin memiliki kelompok pemain yang tidak boleh bermain bersama. Misalnya, jika Anda mempublikasikan game Anda dalam dua bahasa, setiap bahasa harus memiliki server gamenya sendiri.

Untuk mengatur penempatan sesi permainan untuk populasi pemain Anda, buat antrean terpisah untuk setiap segmen pemain. Lingkup setiap antrian untuk menempatkan pemain ke server game yang benar. Beberapa cara umum untuk lingkup antrian meliputi:

- Berdasarkan lokasi geografis. Saat menerapkan server game Anda di beberapa area geografis, Anda dapat membuat antrian untuk pemain di setiap lokasi untuk mengurangi latensi pemain.
- Dengan variasi build atau skrip. Jika Anda memiliki lebih dari satu variasi server game, Anda mungkin mendukung grup pemain yang tidak dapat bermain dalam sesi game yang sama. Misalnya, build atau skrip server game mungkin mendukung berbagai bahasa atau jenis perangkat.
- Berdasarkan jenis acara. Anda dapat membuat antrean khusus untuk mengelola game bagi peserta dalam turnamen atau kejadian khusus lainnya.

Desain beberapa antrian

Tergantung pada permainan dan pemain Anda, Anda mungkin ingin membuat lebih dari satu antrian sesi permainan. Ketika layanan client game Anda meminta sesi game baru, itu menentukan antrean sesi game mana yang akan digunakan. Untuk membantu Anda menentukan apakah akan menggunakan beberapa antrian, pertimbangkan:

- Variasi server game Anda. Anda dapat membuat antrian terpisah untuk setiap variasi server game Anda. Semua armada dalam antrian harus menyebarkan server game yang kompatibel. Ini karena pemain yang menggunakan antrian untuk bergabung dengan game harus dapat bermain di server game antrian mana pun.
- Kelompok pemain yang berbeda. Anda dapat menyesuaikan cara Amazon GameLift Servers menempatkan sesi permainan berdasarkan grup pemain. Misalnya, Anda mungkin memerlukan antrean yang disesuaikan untuk mode game tertentu yang memerlukan tipe instans atau konfigurasi waktu aktif khusus. Atau, Anda mungkin ingin antrian khusus untuk mengelola penempatan untuk turnamen atau acara lainnya.
- Metrik antrian sesi permainan. Anda dapat mengatur antrian berdasarkan cara Anda ingin mengumpulkan metrik penempatan sesi game. Untuk informasi selengkapnya, lihat [Amazon GameLift Serversmetrik untuk antrian](#).

Membangun antrian multi-lokasi

Kami merekomendasikan desain multi-lokasi untuk semua antrian. Desain ini dapat meningkatkan kecepatan penempatan dan ketahanan hosting. Desain multi-lokasi diperlukan untuk menggunakan data latensi pemain untuk menempatkan pemain ke sesi permainan dengan latensi minimal. Jika Anda membangun antrian multi-lokasi yang menggunakan armada Spot Instance, ikuti petunjuknya.

[Tutorial: Membuat Amazon GameLift Servers antrian dengan Instans Spot](#)

Salah satu cara untuk membuat antrian multi-lokasi adalah dengan menambahkan [armada multi-lokasi](#) ke antrian. Dengan begitu, antrian dapat menempatkan sesi permainan di salah satu lokasi armada. Anda juga dapat menambahkan armada lain dengan konfigurasi berbeda atau lokasi rumah untuk redundansi. Jika Anda menggunakan armada Instans Spot multi-lokasi, ikuti praktik terbaik dan sertakan armada Instans Sesuai Permintaan dengan lokasi yang sama.

Contoh berikut menguraikan proses merancang antrian multi-lokasi dasar. Dalam contoh ini, kita menggunakan dua armada: satu armada Instans Spot dan satu armada Instance On-Demand. Setiap armada memiliki yang berikut Wilayah AWS untuk lokasi penempatan: `us-east-1`, `us-east-2`, `ca-central-1`, dan `us-west-2`.

Untuk membuat antrian multi-lokasi dasar dengan armada multi-lokasi

1. Pilih lokasi untuk membuat antrian. Anda dapat meminimalkan latensi permintaan dengan menempatkan antrian di lokasi dekat tempat Anda menerapkan layanan klien. Dalam contoh ini, kita membuat antrian di `us-east-1`.
2. Buat antrian baru dan tambahkan armada multi-lokasi Anda sebagai tujuan antrian. Urutan tujuan menentukan bagaimana Amazon GameLift Servers menempatkan sesi permainan. Dalam contoh ini, kami mencantumkan armada Spot Instance terlebih dahulu dan armada Instance On-Demand kedua.
3. Tentukan urutan prioritas penempatan sesi permainan antrian. Urutan ini menentukan di mana antrian mencari server game yang tersedia terlebih dahulu. Dalam contoh ini, kita menggunakan urutan prioritas default.
4. Tentukan urutan lokasi. Jika Anda tidak menentukan urutan lokasi, Amazon GameLift Servers gunakan lokasi dalam urutan abjad.

Game session placement locations

Locations where the queue can place new game sessions.

Locations

Choose locations ▼

ca-central-1
Canada (Central) ✕

us-west-2
US West (Oregon) ✕

us-east-2
US East (Ohio) ✕

us-east-1
US East (N. Virginia) ✕

Destination order

An ordered list of fleets and aliases that the queue can use for game session placement.

	Region	Type	Name	
⋮	us-east-1 ▼	Fleet ▼	TestFleet-SPOT ▼	Remove
⋮	us-east-1 ▼	Fleet ▼	TestFleet-ONDEMAND ▼	Remove

Add Destination

Game session placement priority

The values that GameLift uses to prioritize game session placement. The default order is latency, cost, destination, and location.

 Latency

Prioritize locations with the lowest average player latency.

 Cost

Prioritize destinations with the lowest current hosting cost.

 Destination

Prioritize based on the defined destination order.

 Location

Prioritize based on the defined location order.

▼ Location order

An ordered list of locations that the queue can use for game session placement.

Location

ca-central-1

▼

Remove

us-east-1

▼

Remove

us-east-2

▼

Remove

us-west-2

▼

Remove

Add locations

Prioritaskan penempatan sesi permainan

Amazon GameLift Servers menggunakan algoritma untuk menentukan bagaimana memprioritaskan tujuan antrian dan menentukan tempat untuk menempatkan sesi permainan baru. Algoritma ini didasarkan pada serangkaian kriteria yang diurutkan. Anda dapat menggunakan urutan prioritas default, atau Anda dapat menyesuaikan pesanan. Anda dapat mengedit urutan prioritas antrian kapan saja.

Urutan prioritas default

1. Latensi — Jika permintaan penempatan sesi permainan menyertakan data latensi spesifik lokasi untuk pemain, Amazon GameLift Servers hitung latensi pemain rata-rata di setiap lokasi dan mencoba menempatkan sesi permainan di lokasi armada dengan rata-rata terendah.
2. Biaya — Jika permintaan tidak menyertakan data latensi, atau jika beberapa armada memiliki latensi yang sama, maka Amazon GameLift Servers evaluasi biaya hosting setiap armada. Biaya hosting armada bervariasi berdasarkan jenis armada (Spot atau On-Demand), jenis instans, dan lokasi.
3. Tujuan - Jika beberapa armada memiliki latensi dan biaya yang sama, maka Amazon GameLift Servers memprioritaskan armada berdasarkan urutan tujuan seperti yang tercantum dalam konfigurasi antrian.
4. Lokasi — Untuk antrian dengan armada multi-lokasi, jika semua kriteria lainnya sama, maka Amazon GameLift Servers prioritaskan lokasi armada berdasarkan urutan abjad.

Sesuaikan cara antrian memprioritaskan penempatan sesi game

Anda dapat memilih untuk menyesuaikan bagaimana antrian memprioritaskan kriteria penempatan. Antrian menerapkan prioritas khusus untuk semua permintaan penempatan sesi game yang diterimanya.

Note

Jika Anda membuat konfigurasi prioritas khusus dan tidak menyertakan keempat kriteria, Amazon GameLift Servers secara otomatis menambahkan kriteria yang hilang dalam urutan default.

Untuk menyesuaikan konfigurasi prioritas antrian

Gunakan [Amazon GameLift Serverskonsol](#) atau AWS Command Line Interface (AWS CLI) untuk membuat konfigurasi prioritas khusus.

Console

Di [Amazon GameLift Serverskonsol](#), Anda dapat menyesuaikan prioritas antrian saat membuat antrean baru atau memperbarui antrian yang ada. Pilih AWS Wilayah untuk bekerja.

Buka bilah navigasi kiri konsol dan pilih Antrian. Pada halaman Antrian, pilih antrian yang ada dan pilih Edit.

1. Pergi ke bagian Prioritas penempatan sesi permainan. Seret dan lepas setiap kriteria prioritas untuk membuat urutan yang Anda inginkan.
2. Pergi ke bagian Urutan lokasi. Tambahkan lokasi apa pun yang ingin Anda prioritaskan. Daftar ini berguna ketika antrian memiliki armada dengan beberapa lokasi. Minimal, Anda harus menentukan satu lokasi. Lokasi yang Anda tentukan di sini diprioritaskan terlebih dahulu, diikuti oleh semua lokasi lain di tujuan antrian.
3. Pilih Simpan perubahan.

AWS CLI

Gunakan [update-game-session-queue](#) perintah dengan `--priority-configuration` opsi untuk menyesuaikan urutan prioritas antrian. Amazon GameLift Servers memperbarui antrian di AWS Wilayah default Anda saat ini, atau Anda dapat menambahkan `--region` tag untuk menentukan AWS Wilayah yang berbeda.

Contoh permintaan berikut menambahkan atau memperbarui konfigurasi prioritas untuk antrian tertentu

```
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" \  
  \
```

Prioritaskan penempatan berdasarkan latensi pemain

Jika Anda ingin memberikan pemain Anda pengalaman pemain terbaik dan memastikan latensi minimal, ambil langkah-langkah berikut saat mengatur sistem penempatan sesi permainan Anda:

- Atur antrian Anda untuk memprioritaskan latensi saat memilih tempat untuk menempatkan sesi permainan. Latensi berada di bagian atas daftar prioritas secara default. Anda juga dapat menyesuaikan konfigurasi prioritas antrian Anda dan memilih tempat untuk menempatkan latensi dalam urutan prioritas.
- Siapkan kebijakan latensi pemain untuk antrian Anda. Kebijakan latensi memungkinkan Anda menetapkan batasan keras pada jumlah latensi yang diizinkan dalam penempatan sesi game.

Jika tidak Amazon GameLift Servers dapat menempatkan sesi permainan tanpa melebihi batas, permintaan penempatan akan habis waktu dan gagal. Anda dapat menyiapkan kebijakan latensi tunggal, atau Anda dapat membuat serangkaian kebijakan yang secara bertahap melonggarkan batas latensi dari waktu ke waktu. Dengan serangkaian kebijakan, Anda dapat menentukan batas latensi awal yang sangat rendah, dan masih mengakomodasi pemain dengan latensi yang lebih tinggi setelah penundaan singkat. Untuk detail tentang membuat kebijakan latensi, lihat [Buat kebijakan latensi pemain](#).

- Saat membuat permintaan penempatan sesi game (lihat [StartGameSessionPlacement](#)), sertakan data latensi untuk setiap pemain. Data latensi pemain mencakup nilai untuk setiap lokasi yang memungkinkan di mana sesi permainan dapat ditempatkan. Misalnya, untuk antrian yang menempatkan sesi game di Wilayah AWS us-east-2 dan ca-central-1, data latensi mungkin terlihat seperti ini:

```
"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" }  
]
```

Untuk mendapatkan pengukuran latensi yang akurat, gunakan ping Amazon GameLift Servers beacon UDP. Titik akhir ini memungkinkan Anda mengukur latensi jaringan UDP aktual antara perangkat pemain dan masing-masing lokasi hosting potensial, menghasilkan keputusan penempatan yang lebih akurat daripada menggunakan ping ICMP. Untuk informasi lebih lanjut tentang penggunaan suar ping UDP untuk mengukur latensi, lihat [Suar ping UDP](#).

Prioritaskan penempatan berdasarkan lokasi

Anda dapat mengonfigurasi antrian untuk membuat penempatan sesi game berdasarkan daftar lokasi geografis yang diprioritaskan. Lokasi adalah salah satu kriteria yang menentukan bagaimana antrian memilih tempat untuk menempatkan sesi permainan baru. Secara default, lokasi diprioritaskan keempat, setelah latensi, biaya, dan tujuan.

Untuk penempatan sesi permainan, tujuan dan lokasi memiliki arti yang agak berbeda:

- Tujuan mengacu pada armada tertentu dan mencakup semua sumber daya hosting armada, di mana pun mereka ditempatkan. Saat memprioritaskan berdasarkan tujuan, Amazon GameLift Servers mungkin membuat penempatan dengan lokasi mana pun di armada. Armada terkelola multi-lokasi dan armada Anywhere dapat memiliki sumber daya hosting yang dikerahkan ke satu atau beberapa lokasi.
- Lokasi mengacu pada posisi geografis tertentu di mana sumber daya hosting armada dikerahkan. Armada dapat memiliki beberapa lokasi, yang mungkin termasuk Wilayah AWS, Local Zones, atau lokasi khusus (untuk armada Anywhere). Armada yang dikelola satu lokasi memiliki satu lokasi dan selalu merupakan Wilayah AWS lokasi. Armada yang dikelola multi-lokasi memiliki Wilayah asal dan dapat memiliki lokasi terpencil. Armada Anywhere memiliki satu atau lebih lokasi khusus.

Saat memprioritaskan penempatan berdasarkan lokasi, Amazon GameLift Servers cari tujuan antrian apa pun yang menyertakan lokasi prioritas dan cari sumber daya hosting yang tersedia. Jika ada beberapa tujuan dengan lokasi prioritas, Amazon GameLift Servers lanjutkan ke kriteria prioritas berikutnya (biaya, latensi, tujuan).

Ada beberapa cara yang dapat mempengaruhi bagaimana lokasi antrian diprioritaskan

- Konfigurasi cara antrian menangani semua permintaan penempatan sesi game:
 - Tambahkan konfigurasi prioritas ke antrian. Konfigurasi prioritas antrian mencakup daftar lokasi yang diurutkan. Anda dapat menentukan satu atau beberapa lokasi untuk diprioritaskan. Daftar ini tidak mengecualikan lokasi mana pun, itu hanya memberi tahu Amazon GameLift Servers di mana harus mencari sumber daya hosting yang tersedia terlebih dahulu. Penggunaan umum untuk daftar lokasi yang dipesan adalah ketika Anda ingin menyalurkan sebagian besar lalu lintas ke satu atau lebih lokasi geografis tertentu dan menggunakan lokasi tambahan sebagai kapasitas cadangan. Tambahkan konfigurasi prioritas dengan menelepon [UpdateGameSessionQueue](#).
 - Tambahkan konfigurasi filter ke antrian. Konfigurasi filter adalah daftar izinkan untuk antrian. Ini memberitahu Amazon GameLift Servers untuk mengabaikan lokasi yang tidak ada dalam daftar ketika mencari sumber daya hosting yang tersedia. Ada dua kegunaan umum untuk konfigurasi filter. Pertama, untuk armada dengan beberapa lokasi, Anda dapat menggunakan filter untuk mengecualikan beberapa lokasi armada. Kedua, Anda mungkin ingin untuk sementara melarang penempatan di lokasi tertentu; misalnya, lokasi mungkin mengalami masalah sementara. Karena Anda dapat memperbarui konfigurasi filter antrian kapan saja, Anda dapat dengan mudah menambah dan menghapus lokasi sesuai kebutuhan. Tambahkan konfigurasi filter dengan menelepon [UpdateGameSessionQueue](#).

- Gunakan instruksi khusus untuk permintaan penempatan individu:
 - Sertakan daftar penggantian prioritas dalam permintaan penempatan sesi game. Anda dapat memberikan daftar prioritas alternatif lokasi dengan [StartGameSessionPlacement](#) permintaan apa pun. Daftar ini secara efektif menggantikan prioritas antrian yang dikonfigurasi untuk lokasi untuk satu permintaan itu saja. Itu tidak memengaruhi permintaan lainnya. Fitur override ini memiliki beberapa persyaratan:
 - Gunakan daftar override hanya dengan antrian yang memiliki konfigurasi prioritas LOCATION sebagai prioritas pertama.
 - Jangan sertakan data latensi pemain dalam permintaan penempatan yang sama. Menyertakan data latensi menyiapkan konflik saat memprioritaskan lokasi yang tidak Amazon GameLift Servers dapat diselesaikan.
 - Putuskan bagaimana Anda Amazon GameLift Servers ingin melanjutkan jika tidak dapat menemukan sumber daya yang tersedia pada daftar penggantian prioritas. Pilih antara kembali ke lokasi antrian lainnya, atau batasi penempatan ke daftar penggantian. Secara default, Amazon GameLift Servers kembali mencoba penempatan di lokasi antrian lainnya.
 - Perbarui konfigurasi filter antrian sesuai kebutuhan, seperti menambahkan lokasi pada daftar penggantian. Daftar penggantian tidak membatalkan daftar filter.

Buat kebijakan latensi pemain

Jika permintaan penempatan Anda menyertakan data latensi pemain, Amazon GameLift Servers temukan sesi permainan di lokasi dengan latensi rata-rata terendah untuk semua pemain. Menempatkan sesi permainan berdasarkan latensi pemain rata-rata Amazon GameLift Servers mencegah menempatkan sebagian besar pemain dalam game dengan latensi tinggi. Namun, Amazon GameLift Servers tetap menempatkan pemain dengan latensi ekstrim. Untuk mengakomodasi pemain ini, buat kebijakan latensi pemain.

Kebijakan latensi pemain Amazon GameLift Servers mencegah menempatkan sesi permainan yang diminta di mana saja pemain dalam permintaan akan mengalami latensi di atas nilai maksimum. Kebijakan latensi pemain juga dapat Amazon GameLift Servers mencegah pencocokan permintaan sesi game dengan pemain latensi yang lebih tinggi.

 Tip

Untuk mengelola aturan khusus latensi, seperti mewajibkan latensi serupa di semua pemain dalam grup, Anda dapat menggunakannya [Amazon GameLift ServersFlexMatch](#) untuk membuat aturan perijodohan berbasis latensi.

Misalnya, pertimbangkan antrian ini dengan batas waktu 5 menit dan kebijakan latensi pemain berikut:

1. Habiskan 120 detik untuk mencari lokasi di mana semua latensi pemain kurang dari 50 milidetik.
2. Habiskan 120 detik untuk mencari lokasi di mana semua latensi pemain kurang dari 100 milidetik.
3. Habiskan waktu antrian yang tersisa hingga batas waktu mencari lokasi di mana semua latensi pemain kurang dari 200 milidetik.

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

Evaluasi metrik antrean

Gunakan metrik untuk mengevaluasi seberapa baik performa antrean Anda. Anda dapat melihat metrik yang terkait dengan antrian di [Amazon GameLift Servers konsol](#) atau di Amazon CloudWatch. Untuk daftar dan deskripsi metrik antrian, lihat. [Amazon GameLift Servers metrik untuk antrian](#)

Metrik antrian dapat memberikan wawasan tentang hal-hal berikut:

- Performa antrian keseluruhan — Metrik antrian menunjukkan seberapa berhasil antrian merespons permintaan penempatan. Metrik ini juga dapat membantu Anda mengidentifikasi kapan dan mengapa penempatan gagal. Untuk antrian dengan armada yang diskalakan secara manual, QueueDepth metrik AverageWaitTime dan dapat menunjukkan kapan Anda harus menyesuaikan kapasitas antrian.
- FleetIQ kinerja algoritma — Untuk permintaan penempatan menggunakan FleetIQ algoritma, metrik menunjukkan seberapa sering algoritma menemukan penempatan sesi permainan yang ideal. Penempatan dapat memprioritaskan penggunaan sumber daya dengan latensi pemain terendah atau sumber daya dengan biaya terendah. Ada juga metrik kesalahan yang mengidentifikasi alasan umum mengapa Amazon GameLift Servers tidak dapat menemukan penempatan yang ideal. Untuk informasi lebih lanjut tentang metrik, lihat [Monitor Amazon GameLift Servers dengan Amazon CloudWatch](#).
- Penempatan spesifik lokasi — Untuk antrian multi-lokasi, metrik menunjukkan penempatan yang berhasil berdasarkan lokasi. Untuk antrian yang menggunakan FleetIQ algoritma, data ini memberikan wawasan yang berguna tentang di mana aktivitas pemain terjadi.

Saat mengevaluasi metrik untuk FleetIQ kinerja algoritma, pertimbangkan tips berikut:

- Untuk melacak tingkat antrian menemukan penempatan yang ideal, gunakan PlacementsSucceeded metrik dalam kombinasi dengan FleetIQ metrik untuk latensi terendah dan harga terendah.
- Untuk meningkatkan tingkat antrian dalam menemukan penempatan yang ideal, tinjau metrik kesalahan berikut:
 - Jika tinggi, sesuaikan penskalaan kapasitas untuk armada antrian. FirstChoiceOutOfCapacity
 - Jika metrik FirstChoiceNotViable kesalahan tinggi, lihat armada Instans Spot Anda. Armada Spot Instance dianggap tidak layak ketika tingkat interupsi untuk jenis instans tertentu terlalu tinggi. Untuk mengatasi masalah ini, ubah antrian untuk menggunakan armada Instance Spot dengan tipe instance yang berbeda. Sebaiknya sertakan armada Instans Spot dengan tipe instans yang berbeda di setiap lokasi.

Desain antrian untuk Instans Spot

Anda dapat memanfaatkan penghematan biaya hosting yang signifikan dengan menggunakan armada Spot. Untuk detail selengkapnya, lihat [Instans Sesuai Permintaan versus Instans Spot](#). Untuk menambahkan armada Spot ke solusi hosting Anda, Anda perlu mengonfigurasi antrian sesi permainan dengan kombinasi armada Spot dan armada Sesuai Permintaan. Amazon GameLift Servers menggunakan antrian selama proses penempatan sesi permainan untuk mencari di beberapa armada dan menemukan host terbaik yang tersedia untuk sesi permainan baru. Topik ini memberikan panduan tentang cara mulai menggunakan armada Spot.

Apakah Anda menggunakan FlexMatch untuk perjodohan? Anda dapat menggunakan langkah-langkah berikut untuk menambahkan armada Spot ke antrian sesi permainan yang ada untuk penempatan perjodohan.

1. Tentukan tujuan untuk antrian sesi permainan Anda.

Mengelola penempatan sesi game dengan antrian adalah praktik terbaik, dan diperlukan saat menggunakan Instans Spot. Karena Instans Spot mungkin tidak selalu tersedia saat Anda membutuhkannya, Anda perlu merancang antrian tangguh yang mencakup armada Spot dan armada Sesuai Permintaan untuk menawarkan kapasitas cadangan. Anda dapat menjaga armada On-Demand Anda diperkecil sampai dibutuhkan. Untuk mendesain antrian Anda, pertimbangkan hal berikut:

- Lokasi — Jika memungkinkan, armada Spot dan armada Sesuai Permintaan Anda harus berada di Wilayah yang sama dengan para pemain. Posisikan sumber daya Spot dan sumber daya Sesuai Permintaan di setiap lokasi yang ingin Anda dukung. Armada multi-lokasi mendukung instans Spot dan On-Demand.
- Jenis instans — Pertimbangkan persyaratan perangkat keras server game Anda dan ketersediaan instance di lokasi yang Anda pilih.

Untuk mencoba antrian yang mengoptimalkan ketersediaan dan ketahanan Spot, lihat [Tutorial: Membuat Amazon GameLift Servers antrian dengan Instans Spot](#). Untuk praktik terbaik desain Spot, lihat [Praktik terbaik untuk Amazon GameLift Servers antrian sesi permainan](#).

2. Buat armada untuk antrian yang dioptimalkan Spot Anda.

Berdasarkan desain antrian Anda, buat armada untuk men-deploy server game Anda ke lokasi yang Anda inginkan dan tipe instans. Lihat [Buat EC2 armada Amazon GameLift Servers terkelola](#) untuk membantu membuat dan mengonfigurasi armada baru.

3. Buat antrian sesi permainan Anda.

Tambahkan tujuan armada, konfigurasi proses penempatan sesi game, dan tentukan prioritas penempatan. Lihat [Buat antrean sesi game](#) untuk bantuan membuat dan mengonfigurasi antrean baru.

4. Perbarui layanan klien game Anda untuk menggunakan antrian.

Saat klien game Anda menggunakan antrean untuk meminta sumber daya, antrian menghindari sumber daya dengan kemungkinan gangguan yang tinggi dan memilih lokasi yang sesuai dengan prioritas yang Anda tetapkan. Untuk bantuan men-deploy penempatan sesi game di klien game Anda, lihat [Buat sesi permainan](#).

5. Perbarui server game Anda untuk menangani gangguan Spot.

AWS dapat mengganggu Instans Spot dengan notifikasi 2 menit, ketika perlu kapasitas kembali. Siapkan server game Anda untuk menangani gangguan guna meminimalkan dampak pemain.

Sebelum AWS merebut kembali Instans Spot, ia mengirimkan pemberitahuan penghentian. Amazon GameLift Servers meneruskan pemberitahuan ke semua proses server yang terpengaruh dengan memanggil Amazon GameLift Servers Fungsi callback SDK server. `onProcessTerminate()` Terapkan callback ini untuk mengakhiri sesi permainan atau memindahkan sesi permainan dan pemain ke instance baru. Lihat bantuan [Menanggapi notifikasi shutdown proses server](#) untuk men-deploy `onProcessTerminate()`.

Note

AWS melakukan segala upaya untuk memberikan notifikasi sebelum merebut kembali sebuah instance, tetapi ada kemungkinan bahwa Instans Spot AWS mengklaim kembali sebelum peringatan tiba. Siapkan server game Anda untuk menangani gangguan yang tidak terduga.

6. Tinjau kinerja armada dan antrian Spot Anda.

Tayang Amazon GameLift Servers metrik di Amazon GameLift Servers konsol atau dengan Amazon CloudWatch untuk meninjau kinerja. Untuk informasi lebih lanjut tentang Amazon GameLift Servers metrik, lihat [Monitor Amazon GameLift Servers dengan Amazon CloudWatch](#). Metrik kunci meliputi:

- Tingkat interupsi — Gunakan `GameSessionInterruptions` metrik `InstanceInterruptions` dan untuk melacak jumlah dan frekuensi interupsi terkait SPOT

untuk instance dan sesi permainan. Sesi permainan yang direklamasi oleh AWS memiliki status `TERMINATED` dan alasan status. `INTERRUPTED`

- Efektivitas antrian — Lacak tingkat keberhasilan penempatan, waktu tunggu rata-rata, dan kedalaman antrian untuk mengonfirmasi bahwa armada Spot tidak memengaruhi kinerja antrian Anda.
- Penggunaan armada — Pantau data tentang instance, sesi permainan, dan sesi pemain. Penggunaan untuk armada On-Demand Anda dapat menjadi indikator bahwa antrian menghindari penempatan ke armada Spot Anda untuk menghindari gangguan.

Praktik terbaik untuk antrean dengan armada Spot

Jika antrian Anda menyertakan armada Spot, siapkan antrian yang tangguh. Ini memanfaatkan penghematan biaya dengan armada Spot sambil meminimalkan efek gangguan sesi permainan. Untuk bantuan dalam membangun armada dan antrian sesi permainan dengan benar untuk digunakan dengan armada Spot, lihat [Tutorial: Membuat Amazon GameLift Servers antrian dengan Instans Spot](#) Untuk informasi selengkapnya tentang instans Spot, lihat [Desain antrian untuk Instans Spot](#).

Selain praktik terbaik umum di bagian sebelumnya, pertimbangkan praktik terbaik khusus Spot ini:

- Buat setidaknya satu armada On-Demand di setiap lokasi. Armada On-Demand menyediakan server game cadangan untuk pemain Anda. Anda dapat menjaga skala armada cadangan Anda tetap kecil hingga ia dibutuhkan, dan menggunakan penskalaan otomatis untuk meningkatkan kapasitas Sesuai Permintaan saat armada Spot tidak tersedia.
- Pilih jenis instans yang berbeda di beberapa armada Spot di suatu lokasi. Jika satu jenis Instans Spot menjadi tidak tersedia untuk sementara, interupsi hanya memengaruhi satu armada Spot di lokasi. Praktik terbaik adalah memilih jenis instans yang tersedia secara luas, dan menggunakan jenis instans dalam keluarga yang sama (misalnya, `m5.large`, `m5.xlarge`, `m5.2xlarge`). Gunakan [Amazon GameLift Servers konsol](#) untuk melihat data harga historis untuk jenis contoh.

Atur notifikasi kejadian untuk penempatan sesi game

Anda dapat menggunakan pemberitahuan acara untuk memantau status permintaan penempatan individu. Kami merekomendasikan untuk menyiapkan pemberitahuan acara untuk semua game dengan aktivitas penempatan volume tinggi.

Terdapat dua pilihan untuk mengatur notifikasi kejadian.

- Memiliki Amazon GameLift Servers mempublikasikan pemberitahuan acara ke topik Amazon Simple Notification Service (Amazon SNS) menggunakan antrian.
- Gunakan EventBridge acara Amazon yang dipublikasikan secara otomatis dan rangkaian alatnya untuk mengelola acara.

Untuk daftar acara penempatan sesi permainan yang dipancarkan oleh Amazon GameLift Servers, lihat [Acara penempatan sesi permainan](#).

Menyiapkan topik SNS

Untuk Amazon GameLift Servers untuk mempublikasikan semua peristiwa yang dihasilkan oleh antrian sesi permainan ke topik, atur bidang target notifikasi ke topik.

Untuk menyiapkan topik SNS untuk Amazon GameLift Servers pemberitahuan peristiwa

1. [Masuk ke AWS Management Console dan buka konsol Amazon SNS di https://console.aws.amazon.com/sns/v3/home](https://console.aws.amazon.com/sns/v3/home).
2. Dari halaman Topik SNS, pilih Buat topik dan ikuti petunjuk untuk membuat topik Anda.
3. Di bawah kebijakan Akses, lakukan hal berikut:
 - a. Pilih metode Advanced.
 - b. Tambahkan bagian tebal berikut dari objek JSON ke kebijakan yang ada.

```
{
  "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. (Opsional) Tambahkan kontrol akses tambahan ke topik dengan menambahkan kondisi ke kebijakan sumber daya.
4. Pilih Buat topik.
5. Setelah Anda membuat topik SNS, tambahkan ke antrian selama pembuatan antrian, atau edit antrian yang ada untuk menambahkannya.

Siapkan topik SNS dengan enkripsi sisi server

Dengan enkripsi sisi server (SSE), Anda dapat menyimpan data sensitif dalam topik terenkripsi. SSE melindungi konten pesan dalam topik Amazon SNS menggunakan kunci yang dikelola AWS Key Management Service di AWS KMS(). Untuk informasi selengkapnya tentang enkripsi sisi server dengan Amazon SNS, lihat [Enkripsi saat istirahat di Panduan Pengembang Layanan Pemberitahuan Sederhana Amazon](#).

Untuk menyiapkan topik SNS dengan enkripsi sisi server, tinjau topik berikut:

- [Membuat kunci](#) dalam Panduan AWS Key Management Service Pengembang
- [Mengaktifkan SSE untuk topik di Panduan](#) Pengembang Layanan Pemberitahuan Sederhana Amazon

Saat membuat kunci KMS Anda, gunakan kebijakan kunci KMS berikut:

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

Mengatur EventBridge

Amazon GameLift Servers secara otomatis memposting semua acara penempatan sesi game ke EventBridge. Dengan EventBridge Anda dapat mengatur aturan agar acara dirutekan ke target untuk diproses. Misalnya, Anda dapat menetapkan aturan untuk merutekan acara `PlacementFulfilled` ke AWS Lambda fungsi yang menangani tugas yang mendahului koneksi ke sesi permainan. Untuk informasi selengkapnya EventBridge, lihat [Apa itu Amazon EventBridge?](#) di Panduan EventBridge Pengguna Amazon.

Berikut ini adalah beberapa contoh EventBridge aturan untuk digunakan dengan Amazon GameLift Servers antrian:

Pertandingan acara dari semua Amazon GameLift Servers mengantre

```
{
  "source": [
    "aws.gamelift"
  ],
  "detail-type": [
    "GameLift Queue Placement Event"
  ]
}
```

Mencocokkan acara dari antrian tertentu

```
{
  "source": [
    "aws.gamelift"
  ],
  "detail-type": [
    "GameLift Queue Placement Event"
  ],
  "resources": [
    "arn:aws:gamelift:your_region:your_account:gamesessionqueue/your_queue_name"
  ]
}
```

Tutorial: Membuat Amazon GameLift Servers antrian dengan Instans Spot

Pengantar

Tutorial ini menjelaskan cara mengatur penempatan sesi game untuk game yang digunakan pada armada Spot berbiaya rendah. Armada spot memerlukan langkah tambahan untuk menjaga ketersediaan server game berkelanjutan untuk pemain Anda.

Audiens yang dituju

Tutorial ini untuk pengembang game yang ingin menggunakan armada Spot untuk meng-host server game khusus atau Amazon GameLift ServersRealtime.

Apa yang akan Anda pelajari

- Tentukan kelompok pemain yang dilayani antrian sesi permainan Anda.
- Bangun infrastruktur armada untuk mendukung lingkup antrean sesi game.
- Tetapkan alias untuk setiap armada untuk membuat ID armada menjadi abstrak.
- Buat antrian, tambahkan armada, dan prioritaskan tempat Amazon GameLift Servers sesi permainan.
- Tambahkan kebijakan latensi pemain untuk membantu meminimalkan masalah latensi.

Prasyarat

Sebelum membuat armada dan antrian untuk penempatan sesi permainan, selesaikan tugas-tugas berikut:

- Ulasan [Cara kerja Amazon GameLift Servers](#).
- [Integrasikan server game Anda dengan Amazon GameLift Servers](#).
- [Unggah build atau Realtime skrip server game Anda](#) ke Amazon GameLift Servers.
- [Rencanakan konfigurasi armada Anda](#).

Langkah 1: Tentukan ruang lingkup antrean Anda

Dalam tutorial ini, kami merancang antrian untuk game yang memiliki satu variasi build server game. Saat peluncuran, kami merilis game di dua lokasi: Asia Pasifik (Seoul) dan Asia Pasifik (Singapura). Karena lokasi-lokasi ini dekat satu sama lain, latensi tidak menjadi masalah bagi pemain kami.

Untuk contoh ini, ada satu segmen pemain, yang berarti kita membuat satu antrian. Di masa depan, ketika kami merilis game di Amerika Utara, kami dapat membuat antrian kedua yang dicakup untuk pemain Amerika Utara.

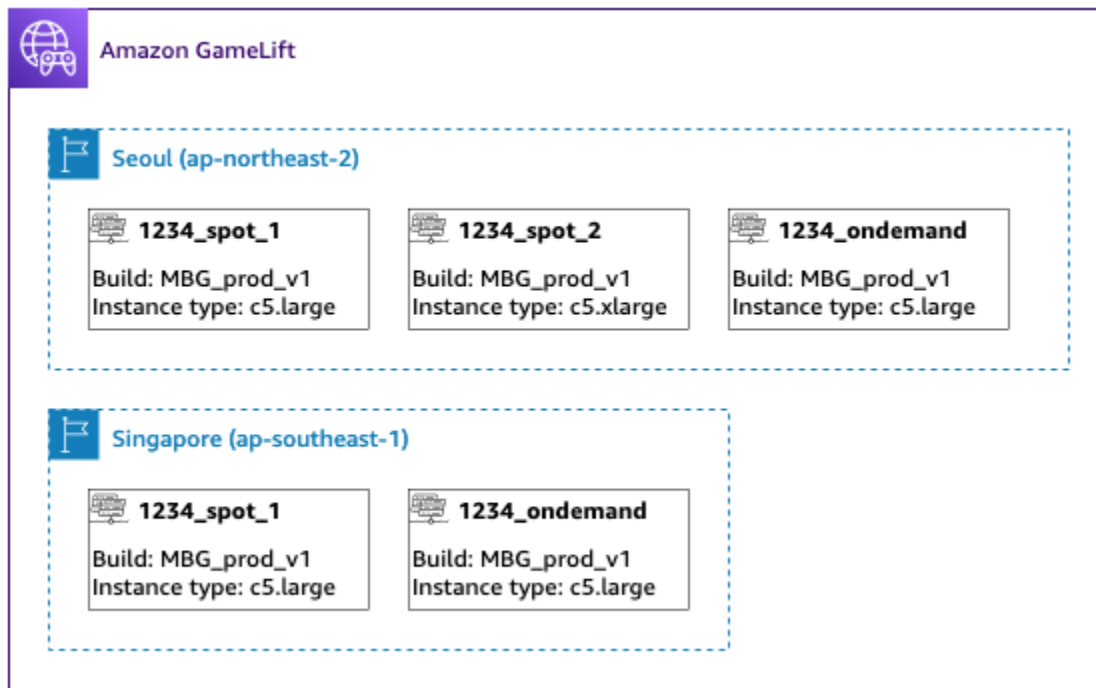
Untuk informasi selengkapnya, lihat [Tentukan ruang lingkup antrian](#).

Langkah 2: Buat Infrastruktur armada Spot

Buat armada di lokasi dan dengan build server game atau skrip yang sesuai dengan cakupan yang Anda tentukan. [Langkah 1: Tentukan ruang lingkup antrean Anda](#)

Dalam tutorial ini, kita membuat infrastruktur dua lokasi dengan setidaknya satu armada Spot dan satu armada On-Demand di setiap lokasi. Setiap armada men-deploy build server game yang sama. Selain itu, kami mengantisipasi bahwa lalu lintas pemain akan lebih berat di lokasi Seoul, jadi kami menambahkan lebih banyak armada Spot di sana.

Diagram berikut menunjukkan contoh infrastruktur armada Spot, dengan 3 armada di lokasi ap-northeast-2 (Seoul) dan 2 armada di lokasi ap-southeast-1 (Singapura). Semua instance di kedua armada menggunakan build MBG_prod_v1. Armada di ap-northeast-2 berisi konfigurasi armada berikut: armada 1234_spot_1 dengan tipe instance c5.large, armada 1234_spot_2 dengan tipe instance c5.xlarge, dan armada 1234_ondemand dengan tipe instance c5.large. Armada di ap-southeast-1 berisi konfigurasi armada berikut: armada 1234_spot_1 dengan tipe instance c5.large dan armada 1234_ondemand dengan tipe instance c5.large.

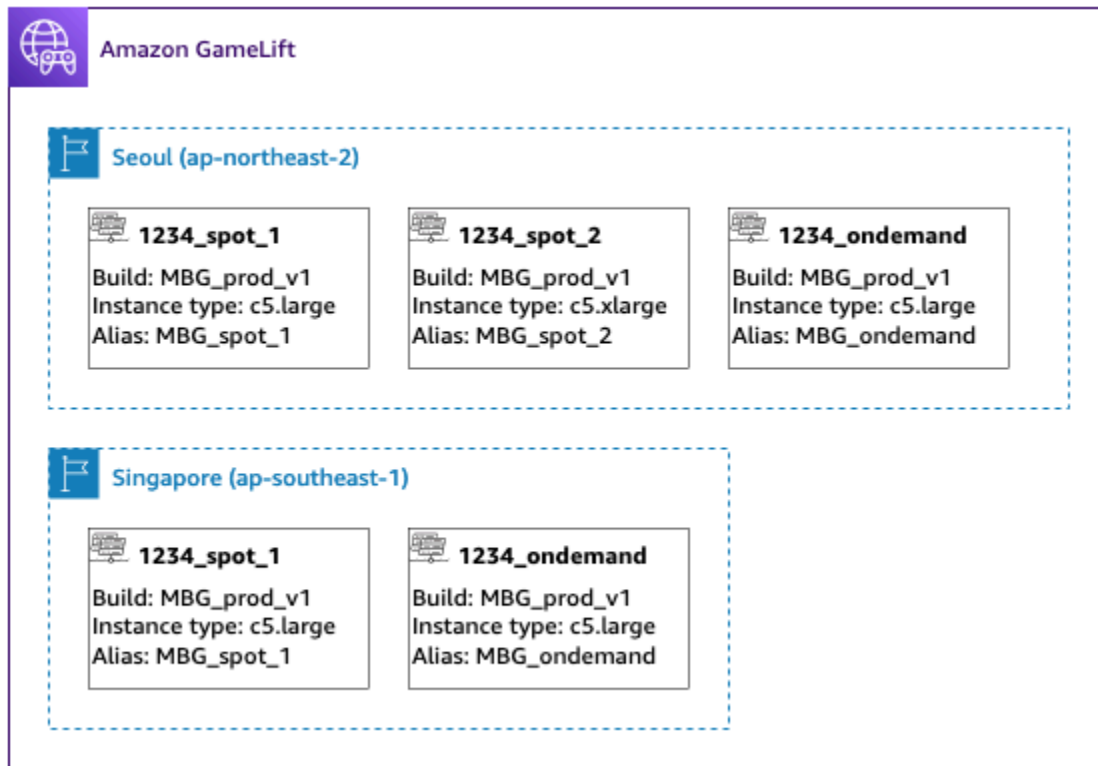


Langkah 3: Tetapkan alias untuk setiap armada

Buat alias baru untuk setiap armada dalam infrastruktur Anda. Alias identitas armada abstrak, membuat penggantian armada berkala menjadi efisien. Untuk informasi selengkapnya tentang membuat alias, lihat [Buat sebuah Amazon GameLift Servers alias](#).

Infrastruktur armada kami memiliki lima armada, jadi kami membuat lima alias menggunakan strategi perutean. Kami membutuhkan tiga alias di lokasi Asia Pasifik (Seoul), dan dua alias di lokasi Asia Pasifik (Singapura).

Diagram berikut menunjukkan infrastruktur armada Spot yang dijelaskan pada langkah kedua dengan alias ditambahkan ke setiap armada. Armada 1234_spot_1 memiliki alias MBG_Spot_1, Armada 1234_spot_2 memiliki alias MBG_Spot_2, dan armada 1234_ondemand memiliki alias MBG_OnDemand.



Untuk informasi selengkapnya, lihat [Membangun antrian multi-lokasi](#).

Langkah 4: Buat antrean dengan tujuan

Buat antrian sesi permainan dan tambahkan tujuan armada Anda. Untuk informasi selengkapnya tentang membuat antrian, lihat [Buat antrean sesi game](#).

Saat membuat antrean Anda:

- Atur batas waktu default menjadi 10 menit. Kemudian, Anda dapat menguji bagaimana batas waktu antrian memengaruhi waktu tunggu pemain Anda untuk masuk ke permainan.
- Lewati bagian kebijakan latensi pemain untuk saat ini. Kami akan membahas ini di langkah berikutnya.
- Prioritaskan armada dalam antrean Anda. Saat bekerja dengan armada Spot, kami merekomendasikan salah satu pendekatan berikut:
 - Jika infrastruktur Anda menggunakan lokasi utama dengan armada di lokasi kedua untuk cadangan, prioritaskan armada terlebih dahulu berdasarkan lokasi kemudian berdasarkan jenis armada.

- Jika infrastruktur Anda menggunakan beberapa lokasi secara merata, prioritaskan armada berdasarkan jenis armada, menempatkan armada Spot di bagian atas antrian.

Untuk tutorial ini, kita membuat antrian baru dengan nama **MBG_spot_queue**, dan menambahkan alias dari kelima armada kita. Kami kemudian memprioritaskan penempatan pertama berdasarkan lokasi dan kedua berdasarkan jenis armada.

Berdasarkan konfigurasi ini, antrian ini selalu mencoba untuk menempatkan sesi permainan baru ke dalam armada Spot di Seoul. Ketika armada itu penuh, antrian menggunakan kapasitas yang tersedia pada armada Seoul On-Demand sebagai cadangan. Jika ketiga armada Seoul tidak tersedia, Amazon GameLift Servers tempatkan sesi permainan di armada Singapura.

Diagram berikut menunjukkan antrian dengan batas waktu 300 detik dan tujuan yang diprioritaskan. Tujuan berada dalam urutan sebagai berikut: 1234_spot_1 di ap-northeast-2, 1234_spot_2 di ap-northeast-2, 1234_ondemand di ap-northeast-2, 1234_ondemand di ap-northeast-2, 1234_spot_1 di ap-southeast-1, dan 1234_ondemand di ap-southeast-1.

 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

Langkah 5: Tambahkan batas latensi pada antrean

Game kami mencakup informasi latensi dalam permintaan penempatan sesi game. Kami juga memiliki fitur pesta pemain yang membuat sesi permainan untuk sekelompok pemain. Kami dapat meminta pemain menunggu sedikit lebih lama untuk masuk ke game dengan pengalaman gameplay yang ideal. Tes permainan kami menunjukkan pengamatan berikut:

- Latensi di bawah 50 milidetik sangat ideal.

- Permainan ini tidak dapat dimainkan pada latensi lebih dari 250 milidetik.
- Pemain menjadi tidak sabar sekitar satu menit.

Untuk antrian kami, dengan batas waktu 300 detik, kami menambahkan pernyataan kebijakan yang membatasi latensi yang diizinkan. Pernyataan kebijakan secara bertahap memungkinkan nilai latensi yang lebih besar hingga latensi 250 milidetik.

Dengan kebijakan ini, antrian kami mencari penempatan dengan latensi ideal (di bawah 50 milidetik) untuk menit pertama, dan kemudian melonggarkan batasnya. Antrian tidak membuat penempatan di mana latensi pemain 250 milidetik atau lebih tinggi.

Diagram berikut menunjukkan antrian dari langkah empat dengan kebijakan latensi pemain ditambahkan. Kebijakan latensi pemain menyatakan, menegakkan batas 50 ms untuk 60-an, menegakkan batas 125 ms selama 30 detik, dan memberlakukan batas 250 ms hingga batas waktu.

 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

Ringkasan

Selamat! Berikut adalah hal-hal yang Anda capai:

- Anda memiliki antrian sesi permainan yang dicakup untuk segmen populasi pemain Anda.

- Antrian Anda menggunakan armada Spot secara efektif dan tangguh saat interupsi Spot terjadi.
- Antrian Anda memprioritaskan armada untuk pengalaman pemain top.
- Antrian memiliki batas latensi untuk melindungi pemain dari pengalaman gameplay yang buruk.

Anda sekarang dapat menggunakan antrian untuk menempatkan sesi permainan untuk pemain yang dilayaninya. Saat membuat permintaan penempatan sesi game untuk para pemain ini, rujuk nama antrian sesi game ini dalam permintaan. Untuk informasi selengkapnya tentang membuat permintaan penempatan sesi game, lihat [Buat sesi permainan](#), atau [Mengintegrasikan klien game untuk Amazon GameLift ServersRealtime](#).

Langkah selanjutnya:

- [Rancang antrian Anda sendiri](#).
- [Buat antrian](#).
- [Gunakan antrian dengan klien game Anda](#).

Menskalakan kapasitas hosting game dengan Amazon GameLift Servers

Kapasitas hosting, diukur dalam contoh, mewakili jumlah sesi permainan yang Amazon GameLift Servers dapat diselenggarakan secara bersamaan dan jumlah pemain bersamaan yang dapat diakomodasi oleh sesi permainan tersebut. Salah satu tugas paling menantang dengan hosting game adalah meningkatkan kapasitas untuk memenuhi permintaan pemain tanpa membuang-buang uang untuk sumber daya yang tidak Anda butuhkan. Untuk informasi selengkapnya, lihat [Penskalaan kapasitas armada](#).

Kapasitas disesuaikan pada tingkat lokasi armada. Semua armada memiliki setidaknya satu lokasi: AWS Wilayah asal armada. Saat melihat atau menskalakan kapasitas, informasi tersebut tercantum berdasarkan lokasi, termasuk Wilayah asal armada dan lokasi terpencil tambahan.

Anda dapat mengatur jumlah instans yang akan dipertahankan secara manual, atau Anda dapat mengatur penskalaan otomatis untuk menyesuaikan kapasitas secara dinamis saat permintaan pemain berubah. Kami menyarankan Anda memulai dengan mengaktifkan opsi penskalaan otomatis berbasis target. Tujuan penskalaan otomatis berbasis target adalah untuk mempertahankan sumber daya hosting yang cukup untuk mengakomodasi pemain saat ini ditambah sedikit tambahan untuk menangani lonjakan permintaan pemain yang tidak terduga. Untuk sebagian besar game, penskalaan otomatis berbasis target menawarkan solusi penskalaan yang sangat efektif.

Anda dapat melakukan sebagian besar aktivitas penskalaan armada menggunakan Amazon GameLift Servers konsol. Anda juga dapat menggunakan AWS SDK atau AWS Command Line Interface (AWS CLI) dengan [API layanan untuk Amazon GameLift Servers](#).

Topik

- [Untuk mengelola kapasitas armada di konsol](#)
- [Tetapkan batas Amazon GameLift Servers kapasitas](#)
- [Atur kapasitas Amazon GameLift Servers armada secara manual](#)
- [Kapasitas armada skala otomatis dengan Amazon GameLift Servers](#)
- [Armada Amazon GameLift Servers kontainer skala](#)

Untuk mengelola kapasitas armada di konsol

1. Buka [konsol Amazon GameLift Servers](#).
2. Di panel navigasi, pilih Hosting, Armada.
3. Pada halaman Armada, pilih nama armada aktif untuk membuka halaman detail armada.
4. Pilih tab Penskalaan. Pada tab ini, Anda dapat:
 - Lihat metrik penskalaan historis untuk seluruh armada.
 - Lihat dan perbarui pengaturan kapasitas untuk setiap lokasi armada, termasuk batas penskalaan dan pengaturan kapasitas saat ini.
 - Perbarui penskalaan otomatis berbasis target, lihat kebijakan penskalaan otomatis berbasis aturan yang diterapkan ke seluruh armada, dan hentikan aktivitas penskalaan otomatis untuk setiap lokasi.

Tetapkan batas Amazon GameLift Servers kapasitas

Saat menskalakan kapasitas hosting untuk lokasi Amazon GameLift Servers armada, baik secara manual atau dengan penskalaan otomatis, pertimbangkan batas penskalaan lokasi. Semua lokasi armada memiliki batas minimum dan maksimum yang menentukan kisaran yang diizinkan untuk kapasitas lokasi. Secara default, batas lokasi armada memiliki minimal 0 instans dan maksimal 1 instans. Sebelum Anda dapat menskalakan lokasi armada, sesuaikan batasnya.

Jika Anda menggunakan penskalaan otomatis, batas maksimum memungkinkan Amazon GameLift Servers untuk meningkatkan lokasi armada untuk memenuhi permintaan pemain tetapi mencegah biaya hosting yang melarikan diri, seperti selama serangan DDOS. Siapkan [CloudWatch alarm Amazon](#) untuk memberi tahu Anda saat kapasitas mendekati batas maksimum, sehingga Anda dapat mengevaluasi situasi dan menyesuaikan secara manual sesuai kebutuhan. (Anda juga dapat [membuat alarm penagihan](#) untuk memantau AWS biaya.) Batas minimum berguna untuk menjaga ketersediaan hosting, bahkan ketika permintaan pemain rendah.

Anda dapat menetapkan batas kapasitas untuk lokasi armada di [Amazon GameLift Servers konsol](#) atau dengan menggunakan AWS Command Line Interface (AWS CLI).

Mengatur batas kapasitas

Console

1. Buka [konsol Amazon GameLift Servers](#).
2. Di panel navigasi, pilih Hosting, Armada.
3. Pada halaman Armada, pilih nama armada aktif untuk membuka halaman detail armada.
4. Pada tab Penskalaan, di bawah Kapasitas penskalaan, pilih lokasi armada, lalu pilih Edit.
5. Dalam kotak dialog Edit kapasitas penskalaan, setel jumlah instans untuk ukuran Min, Instans yang diinginkan, dan ukuran Maks.
6. Pilih Konfirmasi.

AWS CLI

1. Periksa pengaturan kapasitas saat ini. Di jendela baris perintah, gunakan [describe-fleet-location-capacity](#) perintah dengan ID armada dan lokasi yang ingin Anda ubah kapasitasnya. Perintah ini mengembalikan [FleetCapacity](#) objek yang menyertakan pengaturan kapasitas lokasi saat ini. Tentukan apakah batas instans baru dapat mengakomodasi pengaturan instans yang diinginkan saat ini.

```
aws gamelift describe-fleet-location-capacity \
  --fleet-id <fleet identifier> \
  --location <location name>
```

2. Perbarui pengaturan batas. Di jendela baris perintah, gunakan [update-fleet-capacity](#) perintah dengan parameter berikut. Anda dapat menyesuaikan kedua batas instans dan jumlah instans yang diinginkan dengan perintah yang sama.

```
--fleet-id <fleet identifier>
--location <location name>
--max-size <maximum capacity for scaling>
--min-size <minimum capacity for scaling>
--desired-instances <fleet capacity goal>
```

Contoh:

```
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
```

Jika permintaan Anda berhasil, Amazon GameLift Servers kembalikan ID armada. Jika baru max-size atau min-size nilai bertentangan dengan desired-instances pengaturan saat ini, Amazon GameLift Servers mengembalikan kesalahan.

Atur kapasitas Amazon GameLift Servers armada secara manual

Saat Anda membuat armada baru, Amazon GameLift Servers secara otomatis menyetel instance yang diinginkan ke satu instance di setiap lokasi armada. Kemudian, Amazon GameLift Servers menyebarkan satu instance baru di setiap lokasi. Untuk mengubah kapasitas armada, Anda dapat menambahkan kebijakan penskalaan otomatis berbasis target, atau Anda dapat mengatur jumlah instans yang diinginkan untuk lokasi secara manual. Untuk informasi selengkapnya, lihat [Penskalaan kapasitas armada](#).

Menyetel kapasitas armada secara manual dapat berguna ketika Anda tidak memerlukan penskalaan otomatis atau ketika Anda perlu menahan kapasitas pada tingkat tertentu. Pengaturan kapasitas secara manual hanya berfungsi jika Anda tidak menggunakan kebijakan penskalaan otomatis berbasis target. Jika Anda memiliki kebijakan penskalaan otomatis berbasis target, kebijakan tersebut segera mengatur ulang kapasitas yang diinginkan berdasarkan aturan penskalaannya sendiri.

Anda dapat mengatur kapasitas secara manual di Amazon GameLift Servers konsol atau dengan menggunakan AWS Command Line Interface (AWS CLI). Status armada harus aktif.

Tangguhkan penskalaan otomatis

Anda dapat menangguhkan semua aktivitas penskalaan otomatis untuk setiap lokasi armada. Dengan penskalaan otomatis ditangguhkan, jumlah instans yang diinginkan di lokasi armada tetap sama kecuali diubah secara manual. Saat Anda menangguhkan penskalaan otomatis untuk suatu lokasi, hal itu memengaruhi kebijakan armada saat ini dan kebijakan apa pun yang mungkin Anda tentukan di masa mendatang.

Mengatur kapasitas armada secara manual

Console

1. Buka [konsol Amazon GameLift Servers](#).
2. Di panel navigasi, pilih Hosting, Armada.
3. Pada halaman Armada, pilih nama armada aktif untuk membuka halaman detail armada.
4. Pada tab Penskalaan, di bawah Lokasi auto-scaling yang ditangguhkan, pilih setiap lokasi yang ingin ditangguhkan penskalaan otomatis, lalu pilih Tangguhkan.
5. Di bawah Kapasitas penskalaan, pilih lokasi yang ingin Anda atur secara manual, lalu pilih Edit.
6. Di kotak dialog Edit kapasitas penskalaan, tetapkan nilai yang Anda inginkan untuk instans yang diinginkan, lalu pilih Konfirmasi. Ini memberi tahu Amazon GameLift Servers jumlah instance yang harus dipertahankan dalam keadaan aktif, siap untuk menyelenggarakan sesi permainan.

Amazon GameLift Servers menanggapi perubahan dengan menerapkan instance tambahan atau mematikan yang tidak dibutuhkan. Saat Amazon GameLift Servers menyelesaikan proses ini, jumlah instance aktif di lokasi berubah agar sesuai dengan nilai instans yang diinginkan yang diperbarui. Proses ini mungkin memerlukan sedikit waktu.

AWS CLI

1. Periksa pengaturan kapasitas saat ini. Di jendela baris perintah, gunakan [describe-fleet-location-capacity](#) perintah dengan ID armada dan lokasi yang ingin Anda ubah kapasitasnya. Perintah ini mengembalikan [FleetCapacity](#) objek yang menyertakan pengaturan kapasitas lokasi saat ini. Tentukan apakah batas instans dapat mengakomodasi pengaturan instans baru yang diinginkan.

```
aws gamelift describe-fleet-location-capacity \
  --fleet-id <fleet identifier> \
  --location <location name>
```

2. Perbarui kapasitas yang diinginkan. Gunakan [update-fleet-capacity](#) perintah dengan ID armada, lokasi, dan nilai baru untuk instance yang diinginkan. Jika nilai ini berada di luar kisaran batas saat ini, Anda dapat menyesuaikan nilai batas dalam perintah yang sama.

```
--fleet-id <fleet identifier>
```

```
--location <location name>
--desired-instances <fleet capacity as an integer>
--max-size <maximum capacity> [Optional]
--min-size <minimum capacity> [Optional]
```

Contoh:

```
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
```

Jika permintaan Anda berhasil, Amazon GameLift Servers mengembalikan ID armada. Jika pengaturan instance baru yang diinginkan berada di luar batas minimum dan maksimum, Amazon GameLift Servers mengembalikan kesalahan.

Kapasitas armada skala otomatis dengan Amazon GameLift Servers

Gunakan auto-scaling di Amazon GameLift Servers untuk menskalakan kapasitas armada Anda secara dinamis dalam menanggapi aktivitas server game. Saat pemain tiba dan memulai sesi permainan, penskalaan otomatis dapat menambahkan lebih banyak instance; saat permintaan pemain berkurang, penskalaan otomatis dapat menghentikan instance yang tidak dibutuhkan. Penskalaan otomatis adalah cara yang efektif untuk meminimalkan sumber daya dan biaya hosting Anda, sambil tetap memberikan pengalaman pemain yang lancar dan cepat.

Untuk menggunakan penskalaan otomatis, Anda membuat kebijakan penskalaan yang memberi tahu Amazon GameLift Servers kapan harus naik atau turun. Ada dua jenis kebijakan penskalaan: berbasis target dan berbasis aturan. Pendekatan berbasis target — pelacakan target — adalah solusi lengkap. Kami merekomendasikannya sebagai opsi paling sederhana dan paling efektif. Kebijakan penskalaan berbasis aturan mengharuskan Anda untuk menentukan setiap aspek proses pengambilan keputusan penskalaan otomatis, yang berguna untuk mengatasi masalah tertentu. Solusi ini bekerja paling baik sebagai pelengkap penskalaan otomatis berbasis target.

Anda dapat mengelola penskalaan otomatis berbasis target menggunakan Amazon GameLift Servers konsol, AWS Command Line Interface (AWS CLI), atau AWS SDK. Anda dapat mengelola penskalaan otomatis berbasis aturan hanya menggunakan AWS CLI atau AWS SDK, meskipun Anda dapat melihat kebijakan penskalaan berbasis aturan di konsol.

Topik

- [Penskalaan otomatis berbasis target](#)
- [Skala otomatis dengan kebijakan berbasis aturan](#)

Penskalaan otomatis berbasis target

Penskalaan otomatis berbasis target untuk Amazon GameLift Servers menyesuaikan tingkat kapasitas berdasarkan metrik `PercentAvailableGameSessions` armada. Metrik ini mewakili buffer armada yang tersedia untuk peningkatan permintaan pemain secara tiba-tiba.

Alasan utama untuk mempertahankan buffer kapasitas adalah waktu tunggu pemain. Ketika slot sesi permainan siap dan menunggu, dibutuhkan beberapa detik untuk memasukkan pemain baru ke sesi permainan. Jika tidak ada sumber daya yang tersedia, pemain harus menunggu sesi game yang ada berakhir atau agar sumber daya baru tersedia. Diperlukan beberapa menit untuk memulai instance baru dan proses server.

Saat menyiapkan penskalaan otomatis berbasis target, tentukan ukuran buffer yang ingin dipertahankan oleh armada. Karena `PercentAvailableGameSessions` mengukur persentase sumber daya yang tersedia, ukuran buffer sebenarnya adalah persentase dari total kapasitas armada. Amazon GameLift Servers menambah atau menghapus instance untuk mempertahankan ukuran buffer target. Dengan buffer besar, Anda meminimalkan waktu tunggu, tetapi Anda juga membayar sumber daya tambahan yang mungkin tidak Anda gunakan. Jika pemain Anda lebih toleran terhadap waktu tunggu, Anda dapat menurunkan biaya dengan menetapkan buffer kecil.

Untuk mengatur penskalaan otomatis berbasis target

Console

1. Buka [Amazon GameLift Servers konsol](#).
2. Di panel navigasi, pilih Hosting, Armada.
3. Pada halaman Armada, pilih nama armada aktif untuk membuka halaman detail armada.

4. Pilih tab Penskalaan. Tab ini menampilkan metrik penskalaan historis armada dan berisi kendali untuk menyesuaikan pengaturan penskalaan saat ini.
5. Di bawah kapasitas Penskalaan, periksa apakah ukuran Min dan batas ukuran Maks sesuai untuk armada. Dengan auto scaling diaktifkan, kapasitas menyesuaikan antara dua batas ini.
6. Dalam kebijakan auto-scaling berbasis target, pilih Edit.
7. Di kotak dialog Edit kebijakan auto-scaling berbasis target, untuk Persen sesi game yang tersedia, tetapkan persentase yang ingin Anda pertahankan, lalu pilih Konfirmasi. Setelah Anda mengonfirmasi pengaturan, Amazon GameLift Servers menambahkan kebijakan berbasis target baru di bawah kebijakan auto-scaling berbasis Target.

AWS CLI

1. Tetapkan batas kapasitas. Tetapkan nilai batas menggunakan [update-fleet-capacity](#) perintah. Untuk informasi selengkapnya, lihat [Tetapkan batas Amazon GameLift Servers kapasitas](#).
2. Buat kebijakan baru. Buka jendela baris perintah dan gunakan [put-scaling-policy](#) perintah dengan pengaturan parameter kebijakan Anda. Untuk memperbarui kebijakan yang ada, tentukan nama kebijakan dan berikan versi lengkap dari kebijakan yang diperbarui.

```
--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>
```

Contoh:

```
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"
```

Skala otomatis dengan kebijakan berbasis aturan

Kebijakan penskalaan berbasis aturan dalam Amazon GameLift Servers memberikan kontrol halus saat penskalaan otomatis kapasitas armada sebagai respons terhadap aktivitas pemain. Untuk

setiap kebijakan, Anda dapat menautkan penskalaan ke salah satu dari beberapa metrik armada, mengidentifikasi titik pemicu, dan menyesuaikan peristiwa penskalaan atau penurunan skala yang merespons. Kebijakan berbasis aturan berguna untuk melengkapi [penskalaan berbasis target](#) untuk menangani keadaan khusus.

Kebijakan berbasis aturan menyatakan sebagai berikut: “Jika metrik armada memenuhi atau melewati nilai ambang batas untuk jangka waktu tertentu, maka ubah kapasitas armada dengan jumlah tertentu.” Topik ini menjelaskan sintaks yang digunakan untuk membangun pernyataan kebijakan dan memberikan bantuan dengan menciptakan dan mengelola kebijakan berbasis aturan Anda.

Mengelola kebijakan berbasis aturan

Membuat, memperbarui, atau menghapus kebijakan berbasis aturan menggunakan AWS SDK atau AWS Command Line Interface (AWS CLI) dengan [Service](#) API for Amazon GameLift Servers Anda dapat melihat semua kebijakan aktif di Amazon GameLift Servers konsol.

Untuk menghentikan sementara semua kebijakan penskalaan untuk armada, gunakan AWS CLI perintah [stop-fleet-actions](#).

Untuk membuat atau memperbarui kebijakan penskalaan berbasis aturan ():AWS CLI

1. Tetapkan batas kapasitas. Tetapkan salah satu atau kedua nilai batas menggunakan [update-fleet-capacity](#) perintah. Untuk informasi selengkapnya, lihat [Tetapkan batas Amazon GameLift Servers kapasitas](#).
2. Buat kebijakan baru. Buka jendela baris perintah dan gunakan [put-scaling-policy](#) perintah dengan pengaturan parameter kebijakan Anda. Untuk memperbarui kebijakan yang ada, tentukan nama kebijakan dan berikan versi lengkap dari kebijakan yang diperbarui.

```
--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>
```

Contoh:

```
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
```

Untuk menghapus kebijakan penskalaan berbasis aturan menggunakan: AWS CLI

- Buka jendela baris perintah dan gunakan [delete-scaling-policy](#) perintah dengan ID armada dan nama kebijakan.

Contoh:

```
aws gamelift delete-scaling-policy \  
  --fleet-id fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa \  
  --name "Scale up when AGS<50"
```

Sintaks untuk aturan penskalaan otomatis

Untuk membuat pernyataan kebijakan penskalaan berbasis aturan, tentukan enam variabel:

Jika *<metric name>* tetap *<comparison operator>* *<threshold value>* untuk *<evaluation period>*, maka ubah kapasitas armada menggunakan *<adjustment type>* to/by *<adjustment value>*.

Misalnya, pernyataan kebijakan ini memulai acara peningkatan skala setiap kali kapasitas ekstra armada kurang dari yang dibutuhkan untuk menangani 50 sesi permainan baru:

Jika AvailableGameSessions tetap `less than 50` untuk `10 minutes`, maka ubah kapasitas armada menggunakan `ChangeInCapacity` dengan `1 instances`.

Nama metrik

Untuk memulai acara penskalaan, tautkan kebijakan penskalaan otomatis ke salah satu metrik spesifik armada berikut. Untuk deskripsi metrik lengkap, lihat [Amazon GameLift Servers metrik untuk armada](#).

- Mengaktifkan sesi game
- Sesi game aktif
- Sesi game yang tersedia
- Persen sesi game yang tersedia
- Instans aktif
- Sesi pemain yang tersedia
- Sesi pemain saat ini
- Instans Idle
- Persen instans idle

Jika armada berada dalam antrean sesi permainan, Anda dapat menggunakan metrik berikut:

- Kedalaman antrian — Jumlah permintaan sesi permainan yang tertunda armada ini adalah lokasi hosting terbaik yang tersedia.
- Waktu tunggu — Waktu tunggu khusus armada. Lama waktu permintaan sesi game tertunda tertua telah menunggu untuk dipenuhi. Waktu tunggu armada sama dengan waktu permintaan terdahulu saat ini dalam antrean.

Operator perbandingan

Memberitahu Amazon GameLift Servers cara membandingkan data metrik dengan nilai ambang batas. Operator perbandingan yang valid mencakup lebih besar dari (>), kurang dari (> <), greater than or equal (>=), dan kurang dari atau sama (<=).

Nilai ambang batas

Ketika nilai metrik yang ditentukan memenuhi atau melintasi nilai ambang batas, itu memulai peristiwa penskalaan. Nilai ini selalu berupa bilangan bulat positif.

Periode evaluasi

Metrik harus memenuhi atau melewati nilai ambang batas untuk panjang penuh periode evaluasi sebelum memulai acara penskalaan. Panjang periode evaluasi berturut-turut; jika metrik mundur dari ambang batas, periode evaluasi dimulai lagi.

tipe dan nilai penyesuaian

Kumpulan variabel ini bekerja sama untuk menentukan bagaimana Amazon GameLift Servers seharusnya menyesuaikan kapasitas armada saat peristiwa penskalaan dimulai. Pilih dari tiga jenis penyesuaian yang mungkin:

- **Perubahan kapasitas** – Meningkatkan atau mengurangi kapasitas saat ini dengan sejumlah instans tertentu. Tetapkan nilai penyesuaian ke jumlah instans untuk menambah atau menghapusnya dari armada. Nilai-nilai positif menambahkan instans, sementara nilai-nilai negatif menghapus instans. Misalnya, nilai “-10” menurunkan armada sebanyak 10 kali, terlepas dari ukuran total armada.
- **Persen perubahan kapasitas** – Meningkatkan atau mengurangi kapasitas saat ini dengan persentase tertentu. Tetapkan nilai penyesuaian ke persentase yang ingin Anda tambah atau kurangi kapasitas armada. Nilai-nilai positif menambahkan instans, sementara nilai-nilai negatif menghapus instans. Misalnya, untuk armada dengan 50 instance, persentase perubahan “20” menambahkan 10 instance ke armada.
- **Kapasitas yang tepat** — Meningkatkan atau mengurangi kapasitas saat ini ke nilai tertentu. Tetapkan nilai penyesuaian ke jumlah pasti peristiwa yang ingin Anda pertahankan di armada.

Kiat untuk penskalaan otomatis berbasis aturan

Saran berikut dapat membantu Anda mendapatkan hasil maksimal dari penskalaan otomatis dengan kebijakan berbasis aturan.

Menggunakan beberapa kebijakan

Anda dapat memiliki beberapa kebijakan penskalaan otomatis untuk armada secara bersamaan. Skenario yang paling umum adalah memiliki kebijakan berbasis target yang mengelola sebagian large kebutuhan penskalaan dan menggunakan kebijakan berbasis aturan untuk menangani kasus edge. Tidak ada batasan untuk menggunakan beberapa kebijakan.

Dengan berbagai kebijakan, setiap kebijakan berperilaku independen. Tidak ada cara untuk mengontrol urutan peristiwa penskalaan. Misalnya, jika Anda memiliki beberapa kebijakan yang mendorong peningkatan skala, ada kemungkinan aktivitas pemain dapat memulai beberapa peristiwa penskalaan secara bersamaan. Hindari kebijakan yang saling memulai. Misalnya, Anda dapat membuat loop tak terbatas jika Anda membuat kebijakan penskalaan dan penurunan skala yang menetapkan kapasitas di luar ambang batas satu sama lain.

Atur apasitas minimum dan maksimum

Setiap armada memiliki batas kapasitas maksimum dan minimum. Fitur ini penting saat menggunakan penskalaan otomatis. Penskalaan otomatis tidak pernah menetapkan kapasitas ke nilai di luar rentang ini. Secara default, armada yang baru dibuat memiliki nilai minimal 0 dan maksimal 1. Agar kebijakan penskalaan otomatis Anda memengaruhi kapasitas sebagaimana dimaksud, tingkatkan nilai maksimumnya.

Kapasitas armada juga dibatasi oleh batasan jenis instans armada dan kuota layanan di Anda. Akun AWS Anda tidak dapat menetapkan minimum dan maksimum di luar batas dan kuota akun ini.

Melacak metrik setelah perubahan kapasitas

Setelah mengubah kapasitas sebagai respons terhadap kebijakan penskalaan otomatis, Amazon GameLift Servers tunggu 10 menit sebelum menanggapi pemicu dari kebijakan yang sama. Penantian ini memberi Amazon GameLift Servers waktu untuk menambahkan instance baru, meluncurkan server game, menghubungkan pemain, dan mulai mengumpulkan data dari instance baru. Selama waktu ini, Amazon GameLift Servers evaluasi kebijakan terhadap metrik dan melacak periode evaluasi kebijakan, yang dimulai kembali setelah peristiwa penskalaan terjadi. Ini berarti bahwa kebijakan penskalaan dapat memulai acara penskalaan lain segera setelah waktu tunggu selesai.

Tidak ada waktu tunggu antara peristiwa penskalaan yang dimulai oleh kebijakan penskalaan otomatis yang berbeda.

Armada Amazon GameLift Servers kontainer skala

Salah satu tugas paling menantang dengan hosting game adalah meningkatkan kapasitas untuk memenuhi permintaan pemain tanpa membuang-buang uang untuk sumber daya yang tidak Anda butuhkan. Dalam armada kontainer terkelola, Anda menskalakan kapasitas armada Anda dengan menambahkan atau menghapus instance armada.

Saat Anda membuat armada baru, Amazon GameLift Servers tetapkan kapasitas armada yang diinginkan ke satu instans dan sebarkan satu instance di wilayah asal armada. Untuk armada multi-lokasi, Amazon GameLift Servers menyebarkan satu instance ke wilayah asal dan ke setiap lokasi terpencil. Setelah status armada mencapai ACTIVE, Anda dapat meningkatkan kapasitas yang diinginkan untuk menaikkan atau menurunkan kapasitas yang diinginkan untuk menurunkan skala.

Anda dapat menggunakan fitur Amazon GameLift Servers penskalaan untuk mengubah kapasitas secara manual atau mengatur penskalaan otomatis berdasarkan permintaan pemain:

- Siapkan penskalaan otomatis dengan pelacakan target. Lihat [Penskalaan otomatis berbasis target](#).
- Ubah kapasitas armada Anda secara manual. Lihat [Atur kapasitas Amazon GameLift Servers armada secara manual](#).

Saat menskalakan armada kontainer, pertimbangkan bagaimana menambahkan atau menghapus instance memengaruhi kapasitas armada untuk menyelenggarakan sesi permainan dan pemain.

- Sesi permainan per instance
 - Setiap proses server game yang berjalan pada sebuah instance mewakili kapasitas untuk meng-host satu sesi game.
 - Gunakan rumus ini untuk menghitung jumlah sesi permainan yang berjalan secara bersamaan pada instance armada kontainer:

```
[Game sessions per instance] = [# of game server processes per game server container] * [# of game server container groups per instance]
```

Jika arsitektur kontainer Anda menjalankan satu proses server game secara bersamaan di container server game, maka sesi game per instance sama dengan jumlah grup kontainer server game per instance.

- Untuk grup kontainer server game per instance, panggil [DescribeContainerFleet](#) untuk mendapatkan MaximumGameServerContainerGroupsPerInstance nilai GameServerContainerGroupsPerInstance or.
- Pemain per contoh
 - Anda memutuskan jumlah slot pemain untuk memungkinkan di setiap sesi permainan. Bergantung pada bagaimana solusi hosting Anda menangani penempatan sesi game, Anda dapat menentukan pemain per sesi game dalam konfigurasi perjodohan Anda atau dalam panggilan Anda untuk memulai penempatan sesi game.
 - Gunakan rumus ini untuk menghitung jumlah pemain yang dapat memainkan game Anda secara bersamaan pada instance armada kontainer:

```
[Players per instance] = [# of game sessions per instance] * [# of player slots per game session]
```

Untuk mendapatkan total kapasitas armada kontainer saat ini, hubungi [DescribeFleetCapacity](#) atau [DescribeFleetLocation Kapasitas](#) untuk mendapatkan jumlah grup kontainer server game di armada.

Grup aktif adalah mereka yang saat ini menyelenggarakan sesi permainan. Grup idle siap menjadi tuan rumah sesi permainan baru. Lipat gandakan nilai-nilai ini dengan jumlah proses server per grup kontainer server game.

Mempersiapkan game Anda untuk diluncurkan dengan Amazon GameLift Servers hosting

Gunakan daftar periksa berikut untuk memvalidasi setiap fase penerapan game Anda. Item yang ditandai [Critical] sangat penting untuk peluncuran produksi Anda.

[Unduh dan lengkapi kuesioner Amazon GameLift Servers peluncuran, yang tersedia di konsol.](#)

[Amazon GameLift Servers](#) Kami ingin setiap pengembang game memiliki hari peluncuran yang lancar, dan informasi yang diminta membantu kami membantu Anda mempersiapkan pengujian beban, peluncuran lunak, atau peluncuran publik yang akan datang. Amazon GameLift Servers Rencanakan untuk menyerahkan kuesioner yang telah diisi setidaknya tiga (3) bulan sebelum melakukan uji beban pertama Anda.

Topik

- [Siapkan game Anda](#)
- [Bersiaplah untuk pengujian](#)
- [Bersiaplah untuk peluncuran](#)
- [Rencanakan pembaruan pasca-peluncuran](#)

Siapkan game Anda

- [Kritis] Verifikasi bahwa Anda telah menyelesaikan semua [langkah peta jalan pengembangan](#) untuk solusi hosting Anda, dan bahwa Anda memiliki semua komponen yang diperlukan, termasuk dan server game terintegrasi, layanan backend untuk klien game, armada hosting, dan metode penempatan sesi game (seperti antrian).
- [Kritis] [Buat peran AWS Identity and Access Management \(IAM\)](#) yang memungkinkan server game Anda mengakses AWS sumber daya lain saat berjalan.
- [Kritis] Rancang dan terapkan failover ke sumber daya hosting lain sesuai kebutuhan.
- [Rencanakan peluncuran armada ke lokasi target Anda](#), dengan mempertimbangkan antrian permainan dan struktur armada Anda.
- [Otomatiskan penerapan Anda](#) menggunakan infrastruktur sebagai kode (IaC) dengan AWS CloudFormation dan. AWS Cloud Development Kit (AWS CDK)
- [Kumpulkan log dan analitik](#) menggunakan Amazon CloudWatch dan Amazon Simple Storage Service (Amazon S3).

Bersiaplah untuk pengujian

- [Kritis] [Permintaan meningkat untuk kuota Amazon GameLift Servers layanan](#) dan Layanan AWS kuota lainnya sehingga lingkungan hidup Anda dapat meningkatkan kebutuhan produksi.
- [Kritis] Verifikasi bahwa port terbuka pada armada langsung cocok dengan rentang port yang dapat digunakan server Anda.
- [Kritis] Tutup port RDP 3389 dan port SSH 22.
- Kembangkan rencana untuk DevOps pengelolaan game Anda. Jika Anda menggunakan CloudWatch Log Amazon atau metrik CloudWatch kustom Amazon, tentukan alarm untuk masalah serius atau kritis pada armada server. Simulasikan kegagalan dan uji runbook.
- Verifikasi bahwa sumber daya komputasi yang Anda gunakan dapat mendukung jumlah proses server yang ingin Anda jalankan secara bersamaan pada setiap komputasi.
- [Sesuaikan kebijakan penskalaan Anda](#) agar lebih konservatif pada awalnya dan berikan lebih banyak kapasitas menganggur daripada yang Anda pikir Anda butuhkan. Anda dapat mengoptimalkan biaya nanti. Pertimbangkan penggunaan kebijakan penskalaan berbasis target dengan kapasitas menganggur 20 persen.
- Untuk FlexMatch, gunakan [aturan latensi](#) untuk mencocokkan pemain yang secara geografis dekat satu sama lain. Uji perilaku dari aturan ini dengan adanya beban dengan data latensi sintetis dari client uji beban Anda.
- Uji beban otentikasi pemain dan infrastruktur sesi game Anda untuk melihat apakah skalanya efektif untuk memenuhi permintaan.
- Verifikasi bahwa server yang dibiarkan berjalan selama beberapa hari masih dapat menerima koneksi.
- Naikkan level Dukungan rencana Anda ke Bisnis atau Perusahaan sehingga AWS dapat merespons Anda selama masalah atau pemadaman.

Bersiaplah untuk peluncuran

- [Kritis] [Tetapkan kebijakan perlindungan armada](#) ke perlindungan penuh pada semua armada langsung sehingga penskalaan tidak menghentikan sesi permainan aktif.
- [Kritis] [Tetapkan ukuran maksimum armada](#) yang cukup tinggi untuk mengakomodasi permintaan puncak yang diantisipasi, minimal. Kami menyarankan Anda menggandakan ukuran maksimum Anda untuk permintaan yang tidak terduga.

- Dorong seluruh tim pengembangan Anda untuk berpartisipasi dalam acara peluncuran dan pantau peluncuran game Anda di ruang peluncuran.
- Pantau latensi pemain dan pengalaman pemain.

Rencanakan pembaruan pasca-peluncuran

- [Sesuaikan kebijakan penskalaan](#) untuk meminimalkan kapasitas idle berdasarkan penggunaan pemain.
- [Ubah FlexMatch aturan](#) atau [tambahkan lokasi hosting](#) berdasarkan data latensi pemain dan persyaratan yang direvisi.
- Optimalkan konfigurasi runtime untuk menjalankan sesi game sebanyak mungkin di setiap sumber daya komputasi. Mazimisasi efisiensi kinerja dengan cara ini dapat secara langsung memengaruhi biaya armada Anda, karena Anda mungkin dapat menjalankan lebih banyak proses server dengan sumber daya komputasi yang sama.
- [Gunakan data analitik Anda](#) untuk mendorong pengembangan berkelanjutan, meningkatkan pengalaman pemain dan umur panjang game, dan mengoptimalkan monetisasi.

Mengelola Amazon GameLift Servers sumber daya hosting menggunakan AWS CloudFormation

Anda dapat menggunakan AWS CloudFormation untuk mengelola Amazon GameLift Servers sumber daya. Di AWS CloudFormation, Anda membuat template yang memodelkan setiap sumber daya dan kemudian menggunakan template untuk membuat sumber daya Anda. Untuk memperbarui sumber daya, Anda membuat perubahan pada template Anda dan menggunakannya AWS CloudFormation untuk mengimplementasikan pembaruan. Anda dapat mengatur sumber daya Anda ke dalam grup logika, yang disebut tumpukan dan set tumpukan.

Menggunakan AWS CloudFormation untuk mempertahankan Amazon GameLift Servers sumber daya hosting menawarkan cara yang lebih efisien untuk mengelola kumpulan sumber AWS daya. Anda dapat menggunakan kontrol versi untuk melacak perubahan templat dari waktu ke waktu dan mengkoordinasikan pembaruan yang dibuat oleh beberapa anggota tim. Anda juga dapat menggunakan kembali templat. Misalnya, saat men-deploy game di beberapa Wilayah, Anda mungkin menggunakan templat yang sama untuk membuat sumber daya yang sama di setiap Wilayah. Anda juga dapat menggunakan templat ini untuk men-deploy set sumber daya yang sama di partisi lain.

Untuk informasi selengkapnya AWS CloudFormation, lihat [Panduan AWS CloudFormation Pengguna](#). Untuk melihat informasi template untuk Amazon GameLift Servers sumber daya, lihat [Amazon GameLift Servers referensi jenis sumber daya](#).

Praktik terbaik

Untuk panduan mendetail tentang penggunaan AWS CloudFormation, lihat [praktik AWS CloudFormation terbaik](#) di Panduan AWS CloudFormation Pengguna. Selain itu, praktik terbaik ini memiliki relevansi khusus dengan Amazon GameLift Servers.

- Secara konsisten mengelola sumber daya Anda melalui AWS CloudFormation. Jika Anda mengubah sumber daya di luar sumber daya AWS CloudFormation Anda akan tidak sinkron dengan templat sumber daya Anda.
- Gunakan AWS CloudFormation tumpukan dan set tumpukan untuk mengelola beberapa sumber daya secara efisien.
 - Gunakan tumpukan untuk mengelola grup sumber daya yang terhubung. Misalnya, tumpukan yang berisi build, armada yang mereferensikan build, dan alias yang mereferensikan armada.

Jika Anda memperbarui template untuk menggantikan build, AWS CloudFormation ganti armada yang terhubung ke build. AWS CloudFormation kemudian memperbarui alias yang ada untuk menunjuk ke armada baru. Untuk informasi selengkapnya, lihat [Bekerja dengan tumpukan](#) di Panduan AWS CloudFormation Pengguna.

- Gunakan set AWS CloudFormation tumpukan jika Anda menerapkan tumpukan identik di beberapa wilayah atau AWS akun. Untuk informasi selengkapnya, lihat [Bekerja dengan kumpulan tumpukan](#) di Panduan AWS CloudFormation Pengguna.
- Jika Anda menggunakan Instans Spot, sertakan Armada Sesuai Permintaan sebagai cadangan. Kami sarankan Anda mengatur templat Anda dengan dua armada di setiap wilayah, satu armada dengan Instans Spot, dan satu armada dengan Instans Sesuai Permintaan.
- Kelompokkan sumber daya spesifik lokasi dan sumber daya global Anda ke dalam tumpukan terpisah saat Anda mengelola sumber daya di beberapa lokasi.
- Tempatkan sumber daya global Anda dekat dengan layanan yang menggunakannya. Sumber daya seperti antrian dan konfigurasi perjodohan cenderung menerima permintaan volume tinggi dari sumber tertentu. Dengan menempatkan sumber daya Anda dekat dengan sumber permintaan tersebut, Anda meminimalkan waktu perjalanan permintaan dan dapat meningkatkan kinerja secara keseluruhan.
- Tempatkan konfigurasi perjodohan Anda di Wilayah yang sama dengan antrian sesi permainan yang digunakannya.
- Buat alias terpisah untuk setiap armada di tumpukan.

Menggunakan AWS CloudFormation tumpukan

Kami merekomendasikan struktur berikut untuk digunakan saat menyiapkan AWS CloudFormation tumpukan untuk Amazon GameLift Servers sumber daya. Struktur tumpukan optimal Anda bervariasi tergantung pada apakah Anda menyebarkan game Anda di satu lokasi atau beberapa lokasi.

Tumpukan untuk satu lokasi

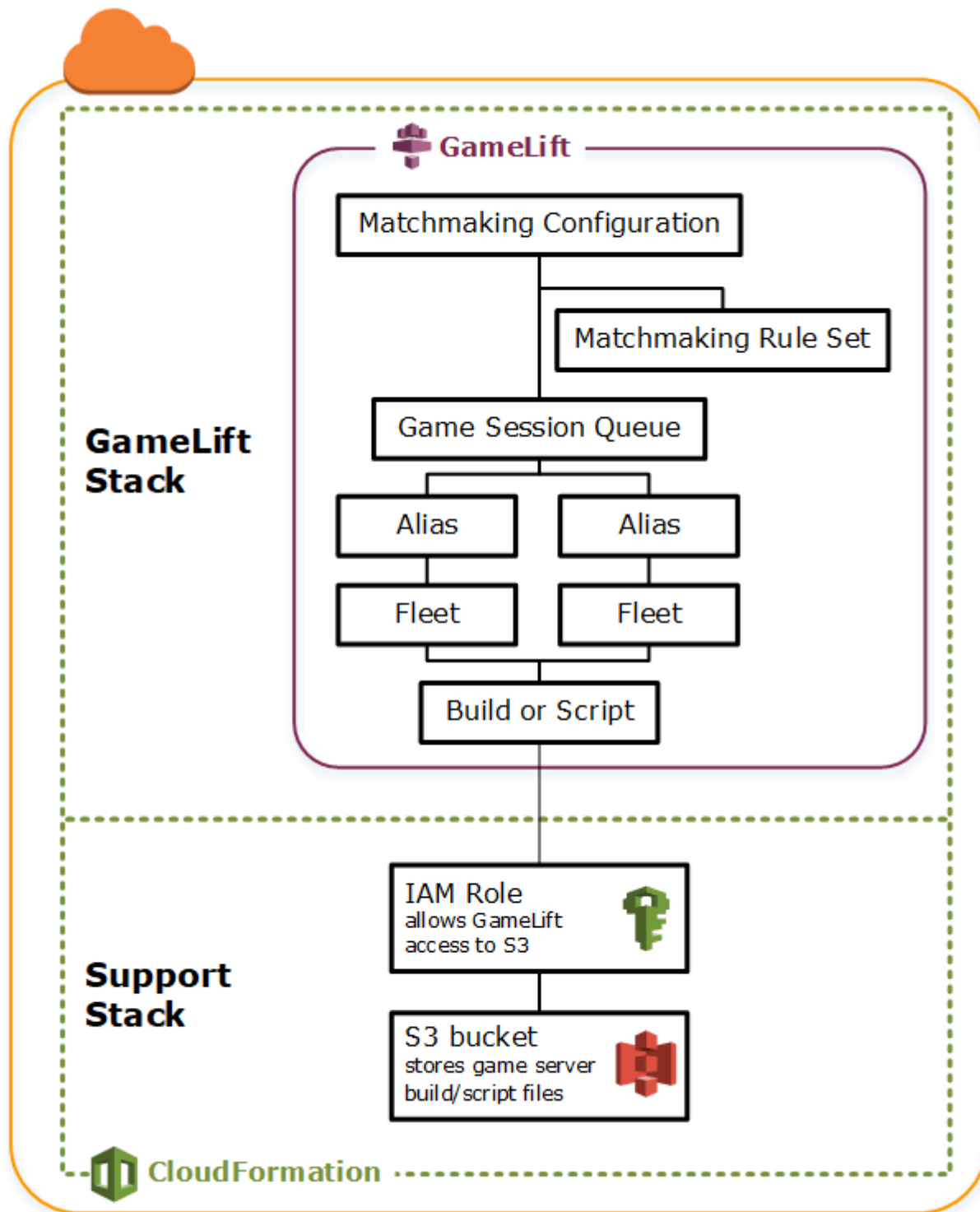
Untuk mengelola Amazon GameLift Servers sumber daya di satu lokasi, kami merekomendasikan struktur dua tumpukan:

- Support Stack - Tumpukan ini berisi sumber daya yang Anda Amazon GameLift Servers sumber daya tergantung pada. Minimal, tumpukan ini harus menyertakan bucket S3 tempat Anda menyimpan server game kustom atau file skrip Realtime. Tumpukan juga harus menyertakan peran

IAM yang memberikan Amazon GameLift Servers izin untuk mengambil file Anda dari bucket S3 saat membuat Amazon GameLift Servers membangun atau sumber daya skrip. Tumpukan ini mungkin juga berisi AWS sumber daya lain yang digunakan dengan game Anda, seperti tabel DynamoDB, cluster Amazon Redshift, dan fungsi Lambda.

- Amazon GameLift Servers stack - Tumpukan ini berisi semua Amazon GameLift Servers sumber daya, termasuk build atau script, satu set armada, alias, dan antrian sesi permainan. AWS CloudFormation membuat sumber daya build atau skrip dengan file yang disimpan di lokasi bucket S3 dan menyebarkan build atau skrip ke satu atau beberapa sumber daya armada. Setiap armada harus memiliki alias yang sesuai. Antrean sesi game mereferensi beberapa atau seluruh alias armada. Jika Anda menggunakan FlexMatch untuk perjodohan, tumpukan ini juga berisi konfigurasi perjodohan dan set aturan.

Diagram di bawah ini menggambarkan struktur dua tumpukan untuk menyebarkan sumber daya dalam satu Wilayah. AWS



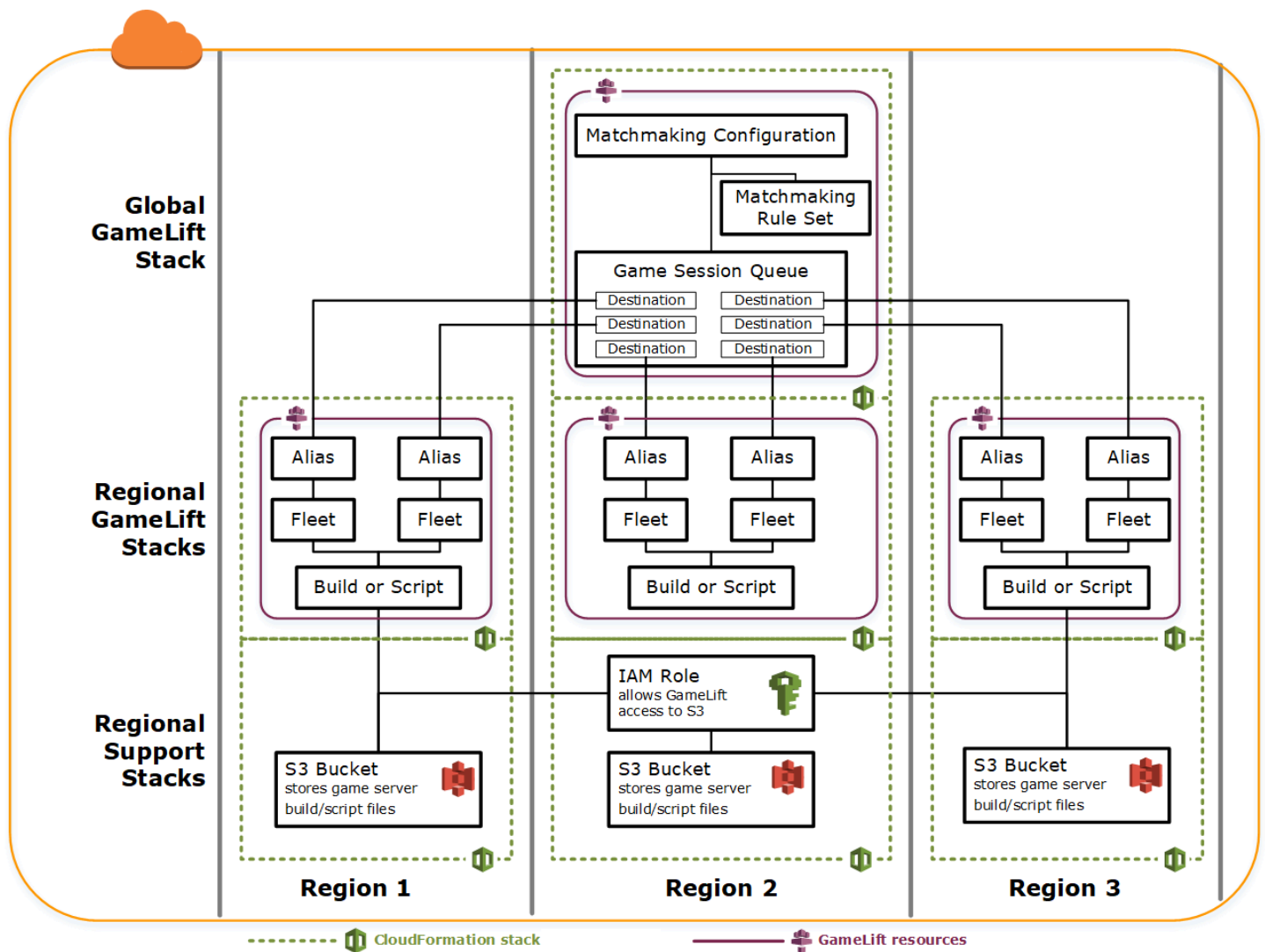
Tumpukan untuk beberapa wilayah

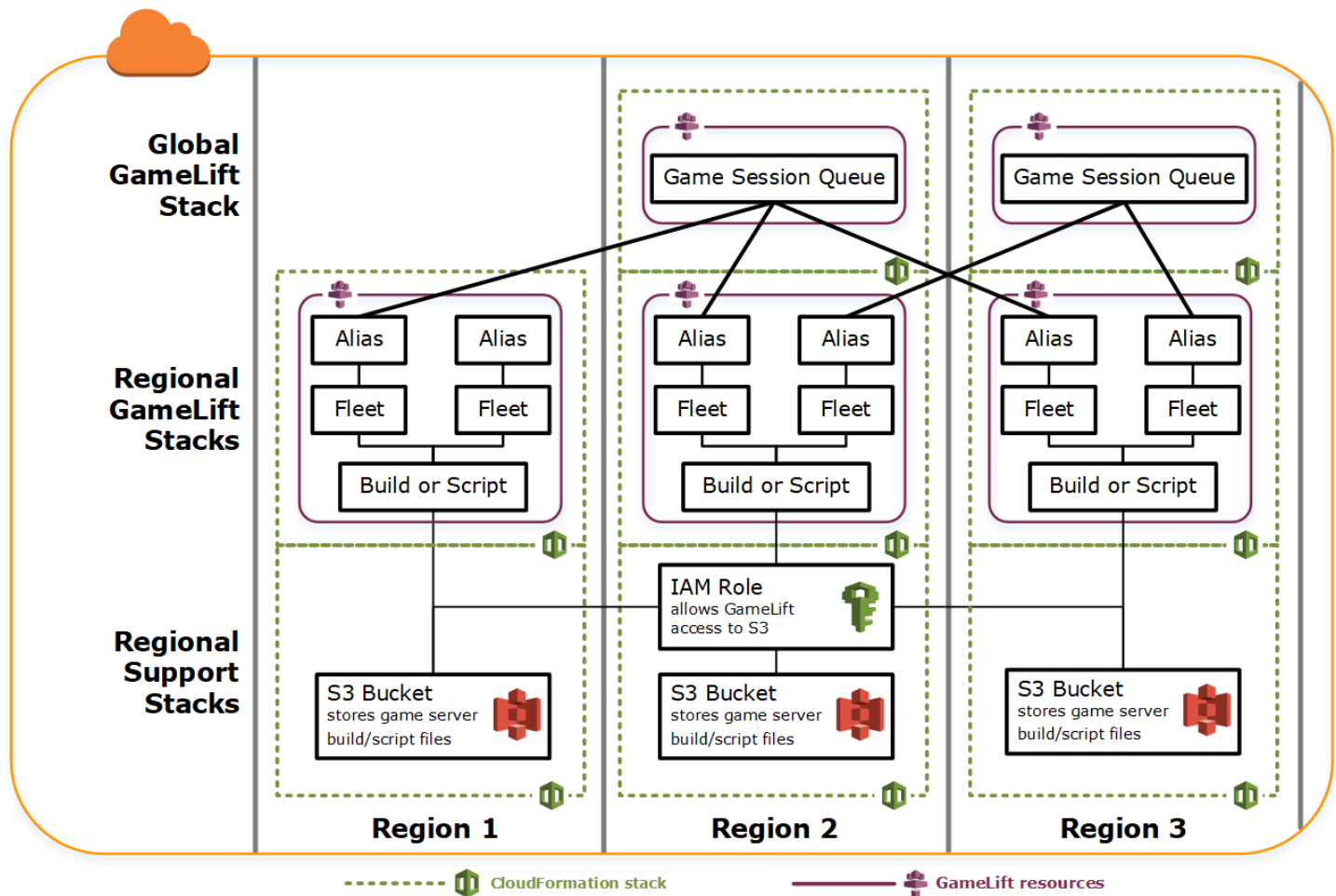
Ketika men-deploy game Anda di lebih dari satu Wilayah, perlu diingat bagaimana sumber daya dapat berinteraksi di seluruh Wilayah. Beberapa sumber daya, seperti Amazon GameLift Servers armada, hanya dapat merujuk sumber daya lain di Wilayah yang sama. Sumber daya lainnya, seperti

Amazon GameLift Servers antrian, adalah Region agnostik. Untuk mengelola Amazon GameLift Servers sumber daya di beberapa Wilayah, kami merekomendasikan struktur berikut.

- **Tumpukan dukungan regional** - Tumpukan ini berisi sumber daya yang Anda Amazon GameLift Servers sumber daya tergantung pada. Tumpukan ini harus mencakup bucket S3 di mana Anda menyimpan server game kustom atau file skrip Realtime. Mungkin juga berisi AWS sumber daya lain untuk game Anda, seperti tabel DynamoDB, kluster Amazon Redshift, dan fungsi Lambda. Banyak dari sumber daya ini adalah khusus Wilayah, sehingga Anda harus membuat sumber daya di setiap Wilayah. Amazon GameLift Servers juga membutuhkan peran IAM yang memungkinkan akses ke sumber daya dukungan ini. Karena IAM role adalah agnostik Wilayah, Anda hanya perlu satu sumber daya peran, ditempatkan di setiap Wilayah dan direferensikan dalam semua tumpukan dukungan lainnya.
- **Regional Amazon GameLift Servers tumpukan** —Tumpukan ini berisi Amazon GameLift Servers sumber daya yang harus ada di setiap wilayah tempat game Anda digunakan, termasuk build atau skrip, satu set armada, dan alias. AWS CloudFormation membuat sumber daya build atau skrip dengan file di lokasi bucket S3, dan menyebarkan build atau skrip ke satu atau beberapa sumber daya armada. Setiap armada harus memiliki alias yang sesuai. Antrean sesi game mereferensi beberapa atau seluruh alias armada. Anda dapat mempertahankan satu templat untuk menggambarkan jenis tumpukan dan menggunakannya untuk membuat set identik sumber daya di setiap Wilayah.
- **Global Amazon GameLift Servers tumpukan** - Tumpukan ini berisi antrian sesi permainan dan sumber daya perijinan Anda. Sumber daya ini dapat ditemukan di Wilayah mana pun dan biasanya ditempatkan di Wilayah yang sama. Antrean dapat mereferensi armada atau alias yang terletak di Wilayah mana pun. Untuk menempatkan antrean tambahan di Wilayah yang berbeda, buat tumpukan global tambahan.

Diagram di bawah ini menggambarkan struktur multistack untuk menyebarkan sumber daya di beberapa Wilayah. AWS Diagram pertama menunjukkan struktur untuk antrean sesi game tunggal. Diagram kedua menunjukkan struktur dengan beberapa antrean.





Memperbarui build

Amazon GameLift Servers build tidak dapat diubah, seperti hubungan antara build dan armada. Akibatnya, ketika Anda memperbarui sumber daya hosting Anda untuk menggunakan satu set baru file build game, hal-hal berikut perlu terjadi:

- Buat build baru menggunakan set file baru (pengganti).
- Buat satu set armada baru untuk men-deploy build game baru (pengganti).
- Alihkan alias untuk menunjuk ke armada baru (update tanpa gangguan).

Untuk informasi selengkapnya, lihat [Memperbarui perilaku sumber daya tumpukan](#) di Panduan AWS CloudFormation Pengguna.

Terapkan pembaruan build secara otomatis

Saat memperbarui tumpukan yang berisi sumber daya build, armada, dan alias terkait, AWS CloudFormation perilaku defaultnya adalah melakukan langkah-langkah ini secara otomatis secara berurutan. Anda memicu pembaruan ini dengan terlebih dahulu mengunggah file build baru ke lokasi S3 baru. Kemudian Anda memodifikasi template AWS CloudFormation build Anda untuk menunjuk ke lokasi S3 baru. Ketika Anda memperbarui tumpukan Anda dengan lokasi S3 baru, ini memicu urutan AWS CloudFormation berikut:

1. Mengambil file baru dari S3, memvalidasi file, dan membuat file baru Amazon GameLift Servers membangun.
2. Memperbarui referensi build dalam templat armada, yang memicu pembuatan armada baru.
3. Setelah armada baru aktif, update referensi armada dalam alias, yang memicu alias untuk memperbarui target armada baru.
4. Menghapus armada lama.
5. Menghapus build lama.

Jika antrean sesi game Anda menggunakan alias armada, lalu lintas pemain secara otomatis beralih ke armada baru segera setelah alias diperbarui. Armada lama secara bertahap ditinggalkan pemain saat sesi game berakhir. Penskalaan otomatis menangani tugas menambahkan dan menghapus instans dari setiap rangkaian armada saat lalu lintas pemain berfluktuasi. Atau, Anda dapat menentukan jumlah instans awal yang diinginkan untuk meningkatkan pengalihan dengan cepat dan mengaktifkan penskalaan otomatis nanti.

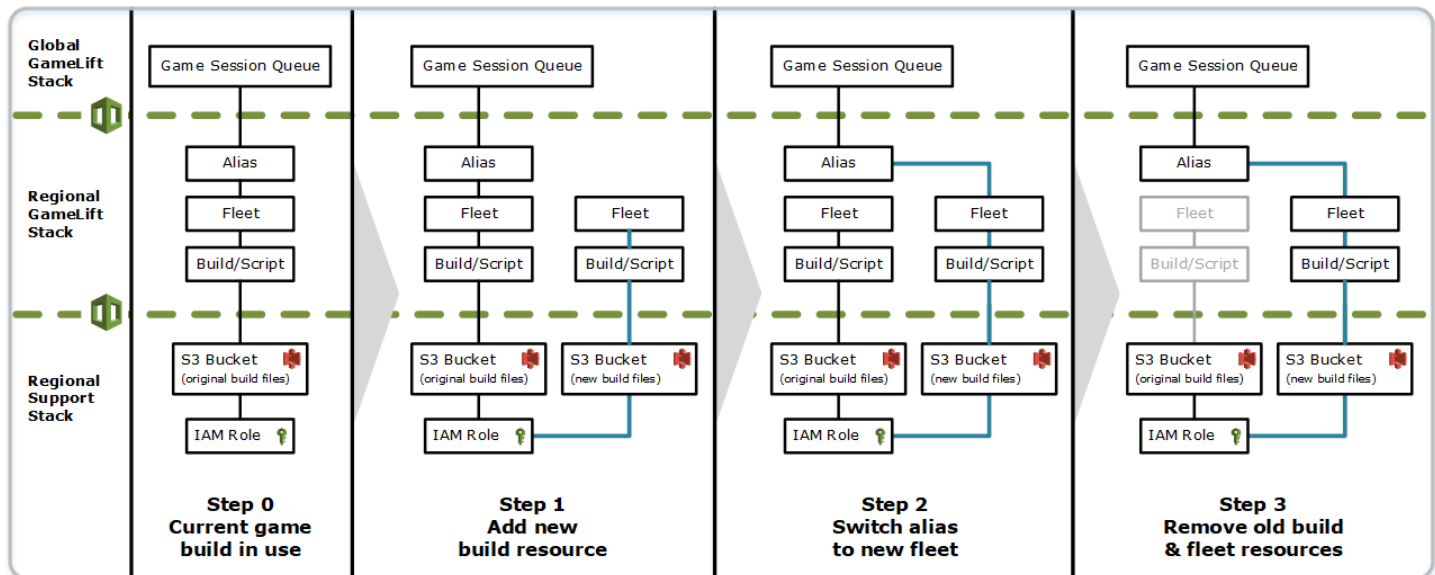
Anda juga dapat AWS CloudFormation menyimpan sumber daya alih-alih menghapusnya. Untuk informasi selengkapnya, lihat [RetainResources](#) di dalam Referensi API AWS CloudFormation .

Terapkan pembaruan build secara manual

Jika Anda ingin memiliki kontrol lebih atas kapan armada baru diluncurkan untuk pemain, Anda memiliki beberapa pilihan. Anda dapat memilih untuk mengelola alias secara manual menggunakan Amazon GameLift Servers konsol atau CLI. Sebagai alternatif, alih-alih memperbarui templat build Anda untuk menggantikan build dan armada, Anda dapat menambahkan set kedua untuk definisi build dan armada ke templat Anda. Saat Anda memperbarui template, AWS CloudFormation buat sumber daya build kedua dan armada yang sesuai. Karena sumber daya yang ada tidak diganti, mereka tidak dihapus, dan alias tetap menunjuk ke armada asli.

Keuntungan utama pendekatan ini adalah pendekatan ini memberi Anda fleksibilitas. Anda dapat membuat sumber daya terpisah untuk versi baru build Anda, menguji sumber daya baru, dan mengontrol ketika armada baru diluncurkan ke pemain. Sebuah kelemahan potensial adalah bahwa dibutuhkan dua kali lebih banyak sumber daya di setiap Wilayah untuk jangka waktu singkat.

Diagram berikut menggambarkan prosesnya.



Bagaimana rollback bekerja

Ketika menjalankan pembaruan sumber daya, jika langkah apapun tidak berhasil diselesaikan, AWS CloudFormation secara otomatis memulai rollback. Proses ini membalikkan setiap langkah secara berurutan, menghapus sumber daya yang baru dibuat.

Jika Anda perlu memicu rollback secara manual, ubah kunci lokasi S3 templat build kembali ke lokasi asli dan perbarui tumpukan Anda. Yang baru Amazon GameLift Servers build dan armada dibuat, dan alias beralih ke armada baru setelah armada aktif. Jika Anda mengelola alias secara terpisah, Anda perlu mengalihkannya untuk menunjuk ke armada baru.

Untuk informasi selengkapnya tentang cara menangani rollback yang gagal atau macet, lihat [Lanjutkan memutar kembali pembaruan](#) di AWS CloudFormation Panduan Pengguna.

Pemantauan Amazon GameLift Servers

Jika Anda menggunakan Amazon GameLift Servers FleetIQ sebagai fitur mandiri dengan Amazon EC2, lihat [Keamanan di Amazon EC2](#) di Panduan EC2 Pengguna Amazon.

Pemantauan merupakan bagian penting dari menjaga keandalan, ketersediaan, dan kinerja Amazon GameLift Servers dan AWS solusi Anda yang lain. Ada tiga kegunaan utama untuk metrik dengan Amazon GameLift Servers: untuk memantau kesehatan sistem dan mengatur alarm, untuk melacak kinerja dan penggunaan server game, dan untuk mengelola kapasitas menggunakan manual atau auto-scaling.

AWS menyediakan alat pemantauan berikut untuk menonton Amazon GameLift Servers, laporkan ketika ada sesuatu yang salah, dan ambil tindakan otomatis bila perlu:

- Amazon GameLift Servers konsol — Gunakan antarmuka grafis untuk mengelola Amazon GameLift Servers sumber daya dan melacak aktivitas hosting game.
- Amazon CloudWatch — Anda dapat memantau Amazon GameLift Servers metrik secara real time, serta metrik untuk AWS sumber daya dan aplikasi lain yang Anda jalankan pada AWS layanan. CloudWatch menawarkan serangkaian fitur pemantauan, termasuk alat untuk membuat dasbor yang disesuaikan dan kemampuan untuk mengatur alarm yang memberi tahu atau mengambil tindakan ketika metrik mencapai ambang batas yang ditentukan.
- AWS CloudTrail— menangkap semua panggilan API dan peristiwa terkait yang dibuat oleh atau atas nama AWS akun Anda Amazon GameLift Servers dan AWS layanan lainnya. Data dikirimkan sebagai berkas log ke bucket Amazon S3 yang Anda tentukan. Anda dapat mengidentifikasi pengguna dan akun mana yang dipanggil AWS, alamat IP sumber dari mana panggilan dilakukan, dan kapan panggilan terjadi.
- Log sesi game - Anda dapat mengeluarkan pesan server khusus untuk sesi game Anda untuk mencatat file yang disimpan di Amazon S3.

Topik

- [Lacak hosting game di Amazon GameLift Servers konsol](#)
- [Monitor Amazon GameLift Servers dengan Amazon CloudWatch](#)
- [Pencatatan log Amazon GameLift Servers Panggilan API dengan AWS CloudTrail](#)
- [Mencatat pesan server di Amazon GameLift Servers](#)

Lacak hosting game di Amazon GameLift Servers konsol

Gunakan Amazon GameLift Servers konsol untuk melihat dan mengelola sumber daya hosting game Anda dan aktivitas hosting yang sedang berlangsung dalam waktu dekat. Konsol menawarkan antarmuka grafis untuk sebagian besar fungsionalitas API layanan untuk Amazon GameLift Servers. Anda dapat menggunakan konsol untuk:

- Gunakan dasbor untuk snapshot tingkat tinggi. Anda dapat melihat angka dan status saat ini dari semua Amazon GameLift Servers hosting sumber daya dan ikuti tautan untuk mendapatkan detail tentang sumber daya individu.
- Kelola sumber daya hosting individu. Anda dapat membuat, melihat, dan menghapus semua Amazon GameLift Servers sumber daya, dan perbarui properti mereka yang bisa berubah. Anda juga dapat melihat jenis aktivitas hosting tertentu, seperti peristiwa dan metrik kinerja.
- Berinteraksi dengan aktivitas permainan dan sesi pemain. Anda dapat melacak sesi permainan dan aktivitas sesi pemain berdasarkan armada, dan menggunakan informasi ini untuk memecahkan masalah sesi game. Lihat detail pada sesi permainan, lihat sesi pemain untuk setiap sesi permainan, dan cari aktivitas pemain di beberapa sesi permainan. Anda juga dapat mematikan sesi permainan individu sesuai kebutuhan.

Topik

- [Dasbor hosting di Amazon GameLift Servers dasbor](#)
- [Server game dibangun di Amazon GameLift Servers dasbor](#)
- [Amazon GameLift ServersRealtimeskrip di konsol Amazon GameLift Servers](#)
- [Armada di Amazon GameLift Servers konsol](#)
- [Detail armada di Amazon GameLift Servers konsol](#)
- [Sesi permainan dan pemain di Amazon GameLift Servers konsol](#)
- [Alias di Amazon GameLift Servers konsol](#)
- [Antrian sesi permainan di Amazon GameLift Servers konsol](#)

Dasbor hosting di Amazon GameLift Servers dasbor

Gunakan Amazon GameLift Servers dasbor konsol untuk mendapatkan snapshot tingkat tinggi tentang status saat ini Amazon GameLift Servers sumber daya hosting di AWS akun Anda. Sebuah Amazon GameLift Servers dashboard memberikan tampilan sebagai berikut:

- Jumlah build dalam status Siap, Inisialisasi, dan Gagal. Pilih Lihat build untuk detail tentang build di Wilayah Anda saat ini.
- Jumlah armada di semua status. Pilih Lihat armada untuk detail tentang armada di Wilayah Anda saat ini.
- Sumber daya Anda saat ini.
- Pengumuman fitur dan layanan baru.

Untuk membuka Amazon GameLift Servers dasbor

- Di [Amazon GameLift Servers konsol](#), di panel navigasi, pilih Dasbor.

Dari dasbor, Anda dapat:

- Persiapkan game Anda untuk diluncurkan dengan memilih Mempersiapkan peluncuran dan mengisi kuesioner peluncuran yang sesuai.
- Permintaan kuota layanan meningkat dalam persiapan peluncuran atau sebagai tanggapan terhadap peluncuran dengan memilih Lihat kuota layanan.
- Lihat posting blog dan informasi rinci tentang fitur baru dengan memilih tautan di sorotan Fitur.



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

 New
0

 Deleting
0

 Error
0

 In progress
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

Server game dibangun di Amazon GameLift Servers dasbor

Pada halaman Builds dari [Amazon GameLift Servers konsol](#), Anda dapat melihat informasi tentang dan mengelola semua build server game yang telah Anda unggah Amazon GameLift Servers untuk penyebaran pada EC2 armada terkelola. Di panel navigasi, pilih Hosting, Managed EC2, Builds.

Halaman Builds menampilkan informasi berikut untuk setiap build. Anda dapat menyesuaikan konten tabel sesuai kebutuhan menggunakan alat Preferensi (lihat



ikon di sudut kanan atas tabel). Preferensi khusus disimpan ke pengguna AWS akun Anda dan diterapkan secara otomatis setiap kali Anda melihat halaman ini.

 Note

Halaman Builds hanya menampilkan build di AWS Wilayah Anda saat ini.

- Nama — Nama yang terkait dengan build yang diunggah.
- Status — Status bangunan. Menampilkan salah satu dari tiga pesan status:
 - Diinisialisasi - Unggahan belum dimulai atau masih dalam proses.
 - Siap - Build siap untuk pembuatan armada.
 - Gagal - Waktu build habis sebelumnya Amazon GameLift Servers menerima binari.
- Waktu pembuatan - Tanggal dan waktu Anda mengunggah build Amazon GameLift Servers.
- Build ID — ID unik yang ditetapkan untuk build saat diunggah.
- Versi — Label versi yang terkait dengan build yang diunggah.
- Sistem operasi — OS yang dijalankan oleh build. OS build menentukan sistem operasi mana Amazon GameLift Servers menginstal pada instance armada.
- Ukuran — Ukuran, dalam megabyte (MB), dari file build yang diunggah ke Amazon GameLift Servers.
- Armada — Jumlah armada yang dikerahkan dengan build.

Dari halaman ini Anda dapat melakukan hal-hal berikut:

- Lihat detail bangunan. Pilih nama build untuk membuka halaman detail build-nya.
- Buat armada baru dari sebuah bangunan. Pilih build, lalu pilih Buat armada.
- Filter dan urutkan daftar bangunan. Gunakan kontrol di bagian atas tabel.
- Menghapus bangunan. Pilih build, lalu pilih Hapus.

Membangun detail

Pada halaman Builds, pilih nama build untuk membuka halaman detailnya. Bagian Ikhtisar pada halaman detail menampilkan informasi ringkasan build yang sama dengan halaman Builds. Bagian

Armada menunjukkan daftar armada yang dibuat dengan build, termasuk informasi ringkasan yang sama dengan halaman [Armada](#).

Amazon GameLift ServersRealtimeskrip di konsol Amazon GameLift Servers

Di halaman Skrip Amazon GameLift Servers [konsol](#), Anda dapat melihat informasi tentang dan mengelola semua Amazon GameLift Servers Realtime skrip yang telah Anda unggah Amazon GameLift Servers untuk penerapan pada armada terkelola. EC2 Di panel navigasi, pilih Hosting, Skrip.

Halaman Skrip menunjukkan informasi berikut untuk setiap skrip. Anda dapat menyesuaikan konten tabel sesuai kebutuhan menggunakan alat Preferensi (lihat



ikon di sudut kanan atas tabel). Preferensi khusus disimpan ke pengguna AWS akun Anda dan diterapkan secara otomatis setiap kali Anda melihat halaman ini.

Note

Halaman Skrip hanya menampilkan skrip di AWS Wilayah Anda saat ini.

- Nama — Nama yang terkait dengan skrip yang diunggah.
- ID - ID unik yang ditetapkan untuk skrip saat diunggah.
- Versi - Label versi yang terkait dengan skrip yang diunggah.
- Ukuran — Ukuran, dalam megabyte (MB), dari file skrip yang diunggah ke. Amazon GameLift Servers
- Waktu pembuatan - Tanggal dan waktu Anda mengunggah skrip. Amazon GameLift Servers
- Armada — Jumlah armada yang dikerahkan dengan skrip.

Dari halaman ini Anda dapat melakukan hal-hal berikut:

- Lihat detail skrip. Pilih nama build untuk membuka halaman detail skripnya.
- Buat armada baru dari skrip. Pilih skrip, lalu pilih Buat armada.
- Filter dan urutkan daftar skrip. Gunakan kontrol di bagian atas tabel.

- Hapus skrip. Pilih skrip, lalu pilih Hapus.

Rincian skrip

Pada halaman Skrip, pilih nama skrip untuk membuka halaman detailnya. Bagian Ikhtisar pada halaman detail menampilkan informasi ringkasan skrip yang sama dengan halaman Builds. Bagian Armada menunjukkan daftar armada yang dibuat dengan skrip, termasuk informasi ringkasan yang sama dengan halaman [Armada](#).

Armada di Amazon GameLift Servers konsol

Anda dapat melihat informasi tentang semua armada yang dibuat untuk meng-host game Anda Amazon GameLift Servers di bawah AWS akun Anda. Daftar ini menunjukkan armada yang membuat Wilayah Anda saat ini. Dari halaman Armada, Anda dapat membuat armada baru atau melihat detail tambahan pada armada. [Halaman detail](#) armada berisi informasi penggunaan, metrik, data sesi permainan, dan data sesi pemain. Anda juga dapat mengedit catatan armada atau menghapus armada.

Untuk melihat halaman Armada, pilih Armada dari panel navigasi.

Halaman Armada menampilkan informasi ringkasan secara default. Anda dapat menyesuaikan konten tabel sesuai kebutuhan menggunakan alat Preferensi (lihat



ikon di sudut kanan atas tabel). Preferensi khusus disimpan ke pengguna AWS akun Anda dan diterapkan secara otomatis setiap kali Anda melihat halaman ini.

- Nama — Nama ramah yang diberikan kepada armada.
- Status – Status armada, yang dapat berupa salah satu dari pernyataan ini: Baru, Mengunduh, Bangunan, dan Aktif.
- Waktu pembuatan - Tanggal dan waktu armada dibuat.
- Jenis komputasi — Jenis komputasi yang digunakan untuk meng-host game Anda. Armada dapat berupa armada yang Dikelola atau EC2 armada Anywhere.
- Jenis instans — Jenis EC2 instans Amazon, yang menentukan kapasitas komputasi instance armada.
- Instance aktif — Jumlah EC2 instans yang digunakan untuk armada.
- Instance yang diinginkan — Jumlah EC2 instans untuk tetap aktif.

- Sesi permainan — Jumlah sesi permainan aktif yang berjalan di armada. Data tertunda lima menit.

Detail armada di Amazon GameLift Servers konsol

Akses halaman detail Armada dari dasbor atau halaman Armada dengan memilih nama armada.

Pada halaman detail armada Anda dapat mengambil tindakan berikut:

- Perbarui atribut armada, pengaturan port, dan konfigurasi waktu aktif.
- Tambahkan atau hapus lokasi armada.
- Ubah pengaturan kapasitas armada.
- Atur atau ubah auto-scaling pelacakan target.
- Hapus armada.

Detail

Pengaturan armada

- ID Armada – Pengenal unik yang ditugaskan untuk armada.
- Nama — Nama armada.
- ARN — Pengenal yang ditugaskan untuk armada ini. ARN armada mengidentifikasinya sebagai Amazon GameLift Servers sumber daya dan menentukan wilayah dan AWS akun.
- Deskripsi — Deskripsi singkat armada yang dapat diidentifikasi.
- Status – Status armada saat ini, yang mungkin saja Baru, Mengunduh, Membangun, dan Aktif.
- Waktu pembuatan - Tanggal dan waktu ketika armada dibuat.
- Waktu penghentian - Tanggal dan waktu armada dihentikan. Ini kosong jika armada masih aktif.
- Tipe Armada – Menunjukkan apakah armada menggunakan sesuai permintaan atau instans spot.
- EC2 type - [Jenis EC2 instans](#) Amazon dipilih untuk armada saat dibuat.
- Peran instans — Peran AWS IAM yang mengelola akses ke AWS sumber daya Anda yang lain, jika peran tersebut diberikan selama pembuatan armada.
- Sertifikat TLS — Apakah armada diaktifkan atau dinonaktifkan untuk menggunakan sertifikat TLS untuk mengautentikasi server game dan mengenkripsi semua komunikasi klien/server.
- Grup metrik — Grup yang digunakan untuk menggabungkan metrik untuk beberapa armada.

- Kebijakan perlindungan penskalaan game — Pengaturan saat ini untuk [perlindungan sesi game](#) untuk armada.
- Sesi permainan maksimum per pemain — Jumlah maksimum sesi yang dapat dibuat pemain selama periode Kebijakan.
- Periode kebijakan — Berapa lama menunggu hingga mengatur ulang jumlah sesi yang telah dibuat pemain.

Membangun detail

Bagian Build details menampilkan build yang dihosting di armada. Pilih nama build untuk melihat halaman detail build lengkap.

Konfigurasi runtime

Bagian konfigurasi Runtime menampilkan proses server untuk diluncurkan pada setiap instance. Ini termasuk jalan untuk server game yang tidak dapat dieksekusi dan opsional parameter peluncuran.

Aktivasi sesi permainan

Bagian aktivasi sesi Game menampilkan jumlah proses server yang diluncurkan pada saat yang sama dan berapa lama menunggu proses diaktifkan sebelum menghentikannya.

EC2 pengaturan port

Bagian Ports menampilkan izin koneksi armada, termasuk alamat IP dan rentang pengaturan port.

Metrik

Tab Metrik menampilkan representasi grafis metrik armada dari waktu ke waktu. Untuk informasi selengkapnya tentang penggunaan metrik di Amazon GameLift Servers, lihat [Monitor Amazon GameLift Servers dengan Amazon CloudWatch](#).

Peristiwa

Tab Peristiwa menyediakan log dari semua peristiwa yang telah terjadi pada armada, termasuk kode peristiwa, pesan, dan stempel waktu. Lihat deskripsi [Acara](#) di Amazon GameLift Servers Referensi API.

Penskalaan

Tab Penskalaan berisi informasi tentang kapasitas armada, termasuk status saat ini dan perubahan kapasitas dari waktu ke waktu. Ini juga menyediakan alat untuk memperbarui batas kapasitas dan mengelola auto-scaling.

Kapasitas penskalaan

Lihat pengaturan kapasitas armada saat ini untuk setiap lokasi armada. Untuk informasi selengkapnya tentang mengubah batas dan kapasitas, lihat [Menskalakan kapasitas hosting game dengan Amazon GameLift Servers](#).

- AWS Lokasi – Nama lokasi tempat instans armada yang tersedia.
- Status – Hosting status lokasi armada. Status lokasi harus ACTIVE untuk dapat menjadi host game.
- Ukuran min - Jumlah instance terkecil yang harus digunakan di lokasi.
- Instance yang diinginkan — Jumlah target instans aktif untuk mempertahankan lokasi. Ketika instans aktif dan instans yang diinginkan tidak sama, peristiwa penskalaan dimulai untuk memulai atau mematikan instance sesuai kebutuhan hingga instance aktif sama dengan instans yang diinginkan.
- Ukuran maksimal - Contoh terbanyak yang dapat digunakan di lokasi.
- Tersedia — Batas layanan pada instans dikurangi jumlah instans yang digunakan. Nilai ini memberitahu Anda jumlah maksimum instans yang dapat Anda tambahkan ke lokasi.

Kebijakan Auto-scaling

Bagian ini mencakup informasi tentang kebijakan auto-scaling yang diterapkan pada armada. Anda dapat menyiapkan atau memperbarui kebijakan berbasis target. Kebijakan berbasis aturan armada, yang harus didefinisikan menggunakan AWS SDK atau CLI, ditampilkan di sini. Untuk informasi selengkapnya tentang penskalaan, lihat [Kapasitas armada skala otomatis dengan Amazon GameLift Servers](#).

Riwayat penskalaan

Lihat grafik perubahan kapasitas dari waktu ke waktu.

Lokasi

Tab Lokasi mencantumkan semua lokasi tempat armada dikerahkan. Lokasi termasuk Wilayah beranda dan lokasi terpencil armada yang telah ditambahkan. Anda dapat menambahkan atau menghapus lokasi langsung di tab ini.

- Lokasi – Nama lokasi tempat instans armada yang tersedia.
- Status – Hosting status lokasi armada. Status lokasi melacak proses pengaktifan instans pertama di lokasi. Status lokasi harus ACTIVE untuk dapat menjadi host games.
- Instance aktif — Jumlah instance dengan proses server yang berjalan di lokasi armada.
- Server aktif — Jumlah proses server game yang dapat meng-host sesi game di lokasi armada.
- Sesi permainan — Jumlah sesi permainan yang aktif pada instance di lokasi armada.
- Sesi pemain — Jumlah sesi pemain, yang mewakili pemain individu, yang berpartisipasi dalam sesi permainan yang aktif di lokasi armada.

Sesi Game

Tab Sesi Game mencantumkan sesi game di masa lalu dan sekarang yang dihosting di armada, termasuk beberapa informasi detail. Pilih ID sesi permainan untuk mengakses informasi sesi permainan tambahan, termasuk sesi pemain. Untuk informasi selengkapnya tentang sesi pemain, lihat [Sesi permainan dan pemain di Amazon GameLift Servers konsol](#).

Sesi permainan dan pemain di Amazon GameLift Servers konsol

Anda dapat menggunakan Amazon GameLift Servers konsol untuk bekerja dengan sesi permainan dan sesi pemain. Untuk informasi lebih lanjut tentang sesi game dan sesi pemain, lihat [Bagaimana pemain terhubung ke game](#). Amazon GameLift ServersKonsol menyediakan informasi dan alat untuk membantu Anda menyelidiki masalah dengan sesi permainan Anda.

Apa yang dapat Anda lakukan:

- Jelajahi sesi permainan dan aktivitas sesi pemain yang diselenggarakan pada armada tertentu.
- Cari aktivitas sesi permainan untuk pemain tertentu di beberapa armada.
- Matikan sesi permainan tertentu.

Lihat detail sesi permainan

Sesi permainan dan data sesi pemain diatur oleh armada yang menyelenggarakan sesi permainan.

Untuk mengakses sesi permainan dan informasi sesi pemain

1. Di [Amazon GameLift Serverskonsol](#), buka panel navigasi kiri. Pilih jenis solusi hosting dan buka halaman Armada. Misalnya:
 - Hosting, Di Mana Saja, Armada
 - Hosting, Dikelola EC2, Armada
 - Hosting, Kontainer terkelola, Armada
2. Setiap halaman Armada menampilkan daftar armada untuk yang Anda pilih saat ini. Wilayah AWS Pilih armada yang ingin Anda lihat data sesi game.
3. Di halaman detail armada, buka tab Sesi permainan. Tab ini mencantumkan semua sesi permainan yang diselenggarakan di armada, bersama dengan informasi ringkasan. Anda dapat menyesuaikan konten tabel sesuai kebutuhan menggunakan alat Preferensi (lihat  ikon di sudut kanan atas tabel). Preferensi khusus disimpan ke pengguna AWS akun Anda dan diterapkan secara otomatis setiap kali Anda melihat halaman ini.
4. Pilih sesi permainan dari daftar untuk melihat informasi tambahan.
5. Jika sesi permainan menyertakan data sesi pemain, pilih Lihat sesi pemain untuk membuka alat pencarian sesi pemain dengan ID sesi permainan yang diisi secara otomatis.

Detail sesi Game mencakup informasi berikut:

- Status – Status sesi game.
 - Mengaktifkan — Instans memulai sesi permainan.
 - Aktif — Sesi permainan sedang berjalan dan tersedia untuk menerima pemain, tergantung pada [kebijakan pembuatan pemain](#) sesi.
 - Dihentikan — sesi permainan telah berakhir.
- ARN — Nama Sumber Daya Amazon dari sesi permainan.
- Nama — Nama yang dihasilkan untuk sesi permainan.
- Lokasi — Lokasi yang Amazon GameLift Servers menjadi tuan rumah sesi permainan di.
- Waktu pembuatan - Tanggal dan waktu yang Amazon GameLift Servers membuat sesi streaming.

- Waktu akhir — Tanggal dan waktu sesi permainan berakhir.
- Nama DNS — Nama host dari sesi permainan.
- Alamat IP – Alamat IP yang ditentukan untuk sesi game.
- Port – Nomor port yang digunakan untuk connect ke sesi game.
- Creator ID — Pengenal unik pemain yang memulai sesi permainan.
- Kebijakan pembuatan sesi pemain — Menunjukkan apakah sesi permainan menerima pemain baru.
- Kebijakan perlindungan penskalaan game — Jenis perlindungan sesi game untuk diatur pada semua instance baru yang Amazon GameLift Servers dimulai di armada.

Data permainan

Data properti game, diformat sebagai string, untuk dikirim ke sesi permainan Anda saat memulai.

Properti game

Data properti game, diformat sebagai key/value pasangan, untuk dikirim ke sesi permainan Anda saat memulai.

Data perjodohan

Jika sesi permainan dibuat dengan FlexMatch, data perjodohan menjelaskan informasi tentang konfigurasi perjodohan dan kumpulan aturan. Ini termasuk atribut pemain masing-masing pertandingan dan tugas tim. Data dalam format JSON. Untuk informasi selengkapnya tentang FlexMatch perjodohan, lihat [Membangun mak comblang](#).

Cari data sesi pemain

Jika solusi hosting game Anda menggunakan sesi pemain dan menyediakan pemain unik IDs, Anda dapat menjelajahi aktivitas khusus pemain untuk sesi permainan sebelumnya atau sekarang di beberapa armada. Buka alat pencarian sesi Pemain menggunakan salah satu metode berikut:

- Di Amazon GameLift Servers konsol, buka panel navigasi kiri dan pilih Pencarian sesi pemain dan pilih jenis filter yang akan digunakan.
- Saat melihat detail sesi permainan armada, pilih Lihat sesi pemain. Alat pencarian terbuka dengan sesi game dengan filter ID sesi game yang telah dipilih sebelumnya dan nilai sesi game diisi.

Saat menggunakan alat pencarian, Anda dapat memberikan informasi berikut:

- ID sesi pemain untuk mendapatkan informasi tentang sesi pemain tertentu.
- ID sesi permainan untuk mendapatkan informasi tentang semua sesi pemain untuk sesi permainan yang diminta. Hasilnya mewakili semua pemain yang memesan slot atau terhubung ke sesi permainan. Anda dapat secara opsional memfilter hasil berdasarkan status sesi pemain.
- ID pemain untuk mendapatkan informasi tentang semua sesi pemain untuk pemain yang diminta. Hasilnya mewakili semua sesi permainan yang diikuti pemain.

 Note

Alat pencarian mencari semua aktivitas sesi pemain di seluruh yang dipilih saat ini. Wilayah AWS Jika Anda memiliki beberapa armada di Wilayah, hasilnya mencakup aktivitas sesi pemain di semua armada. Untuk armada multi-lokasi, hasilnya juga mencakup aktivitas sesi pemain di lokasi terpencil armada.

Data sesi pemain berikut dikumpulkan untuk setiap sesi game:

- ID sesi pemain — Pengidentifikasi yang ditetapkan untuk sesi pemain.
- ID Pemain – Pengidentifikasi unik untuk pemain. Pilih ID ini untuk mendapatkan informasi pemain tambahan.
- ID sesi game — Pengenal yang ditetapkan untuk sesi permainan.
- Fleet ID — Pengenal yang ditugaskan ke armada yang menjadi tuan rumah sesi permainan.
- Status – Status dari sesi pemain. Berikut ini adalah status yang mungkin:
 - Cadangan — Sesi pemain telah dicadangkan, tetapi para pemain tidak terhubung.
 - Aktif — Sesi pemain terhubung ke server game.
 - Selesai – Sesi pemain telah berakhir; pemain tidak lagi terhubung.
 - Timed Out — Pemain gagal terhubung.
- Waktu pembuatan — Waktu pemain terhubung ke sesi permainan.
- Waktu akhir — Waktu pemain terputus dari sesi permainan.
- Data koneksi — Alamat IP, nama DNS dan port yang digunakan pemain untuk terhubung ke sesi permainan.

- Data pemain — Informasi tentang pemain yang disediakan selama pembuatan sesi pemain.

Matikan sesi permainan

Gunakan Amazon GameLift Servers konsol untuk mematikan sesi permainan tertentu. Fitur ini memberi Anda metode sederhana dan cepat untuk menemukan sesi permainan dan mengirim sinyal untuk menghentikannya. Metode penghentian lain mengharuskan Anda menemukan instance armada tempat sesi permainan berjalan, mengakses instance dari jarak jauh, dan mematikan sesi game secara manual.

Anda dapat menutup sesi permainan dengan alasan apa pun. Alasan paling umum adalah menyelesaikan sesi permainan yang gagal ditutup secara alami. Akibatnya, sumber daya hosting untuk sesi game tidak dapat dibebaskan untuk menjadi tuan rumah sesi permainan baru, dan kapasitas hosting armada terdegradasi.

Note

Fitur ini bergantung pada pengaturan konfigurasi tertentu untuk solusi hosting Anda. Ini memiliki keterbatasan sebagai berikut:

- Sesi game harus di-host pada armada yang menjalankan server game yang dibangun dengan server SDK untuk Amazon GameLift Servers v5 atau lebih tinggi. Jika server game Anda digunakan dengan versi yang lebih lama, Anda perlu menggunakan akses jarak jauh untuk menghapus sesi permainan.
- Jika sesi permainan di-host di armada Anywhere, armada harus menggunakan Amazon GameLift Servers Agen untuk mengelola proses server game.

Untuk mengakhiri sesi permainan

1. Di [Amazon GameLift Servers konsol](#), buka panel navigasi kiri. Pilih jenis solusi hosting dan buka halaman Armada. Misalnya:
 - Hosting, Di Mana Saja, Armada
 - Hosting, Dikelola EC2, Armada
 - Hosting, Kontainer terkelola, Armada
2. Setiap halaman Armada menampilkan daftar armada untuk yang Anda pilih saat ini. Wilayah AWS Pilih armada yang menjadi tuan rumah sesi permainan yang ingin Anda hentikan.

3. Di halaman detail armada, buka tab Sesi permainan. Dalam daftar sesi permainan, pilih salah satu yang ingin Anda akhiri, dan pilih tombol Terminate.
4. Dalam sesi permainan Terminate? jendela, verifikasi bahwa Anda mematikan sesi permainan yang tepat dan memilih metode penghentian.
 - Pematian sesi permainan normal — Opsi ini mengirimkan sinyal ke proses server yang menjadi tuan rumah sesi permainan untuk dimatikan. Jika build server game Anda terintegrasi dengan benar Amazon GameLift Servers, proses server memulai urutan shutdown sesi game, memberi tahu Amazon GameLift Servers bahwa itu berakhir, dan berhenti. Tergantung pada desain game Anda, urutan shutdown mungkin mencakup langkah-langkah untuk menyelesaikan sesi permainan dengan anggun, seperti menyimpan data dan memberi tahu pemain aktif. Metode ini mungkin memerlukan sedikit penundaan untuk menyelesaikan urutan shutdown sesi permainan.
 - Shutdown sesi permainan segera - Opsi ini mengirimkan sinyal ke manajer proses untuk mematikan proses server yang menjadi tuan rumah sesi permainan. Opsi ini melewati shutdown sesi permainan normal. Itu dapat mengakhiri sesi permainan bahkan ketika proses server tidak dapat merespons.
5. Konfirmasikan penghentian sesi permainan. Anda dapat melacak progres shutdown di halaman konsol sesi Game. Status sesi permainan akan berubah menjadi “Terminating”, dan kemudian menjadi “Terminated” saat shutdown selesai.

Topik terkait

- Anda juga dapat mematikan sesi game menggunakan AWS SDK dan file. AWS CLI Untuk detail dan contoh selengkapnya, lihat topik Referensi Amazon GameLift Servers API [TerminateGameSession](#).
- Untuk informasi selengkapnya tentang integrasi server game dan bagaimana proses server merespons sinyal dari Amazon GameLift Servers layanan, lihat [Tambahkan Amazon GameLift Servers ke server game Anda](#).

Alias di Amazon GameLift Servers konsol

Halaman Alias menampilkan informasi tentang Amazon GameLift Servers alias yang mengarahkan lalu lintas tujuan hosting tertentu. Untuk alias, pilih Hosting, Alias di panel navigasi.

Anda dapat melakukan hal berikut di halaman alias:

- Buat alias baru. Pilih Buat alias.
- Filter dan urutkan tabel alias. Gunakan kontrol di bagian atas tabel. Anda dapat menyesuaikan konten tabel sesuai kebutuhan menggunakan alat Preferensi (lihat  ikon di sudut kanan atas tabel). Preferensi khusus disimpan ke pengguna AWS akun Anda dan diterapkan secara otomatis setiap kali Anda melihat halaman ini.
- Melihat detail alias. Pilih nama alias untuk membuka halaman detail alias.
- Menghapus alias. Pilih alias dan kemudian pilih Hapus.

Rincian alias

Halaman detail alias menampilkan informasi tentang alias.

Dari halaman ini Anda dapat:

- Edit alias. Pilih Edit.
- Lihat armada yang Anda kaitkan dengan alias.
- Menghapus alias. Pilih Hapus.

Informasi detail alias meliputi:

- ID — Nomor unik yang digunakan untuk mengidentifikasi alias.
- Deskripsi – Deskripsi fungsi.
- ARN — Nama Sumber Daya Amazon dari alias.
- Penciptaan — Tanggal dan waktu alias dibuat.
- Terakhir diperbarui - Tanggal dan waktu alias terakhir diperbarui.
- Jenis routing — Jenis routing untuk alias, yang bisa menjadi salah satunya:
 - Sederhana — Rutekan lalu lintas pemain ke ID armada tertentu. Anda dapat memperbarui ID armada untuk alias kapan saja.
 - Terminal — Mengirimkan pesan kembali ke klien. Misalnya, Anda dapat mengarahkan pemain yang menggunakan out-of-date klien ke lokasi di mana mereka bisa mendapatkan peningkatan.
- Tag — Pasangan kunci dan nilai yang digunakan untuk mengidentifikasi alias.

Antrian sesi permainan di Amazon GameLift Servers konsol

Anda dapat melihat informasi tentang semua antrian, yang digunakan untuk memproses permintaan untuk sesi permainan baru. Halaman antrian menunjukkan antrian sesi permainan dalam yang saat ini dipilih. Wilayah AWS Dari halaman Antrean, Anda dapat membuat antrean baru, menghapus antrean yang ada, atau membuka halaman detail untuk antrean yang dipilih. Setiap halaman detail antrian berisi konfigurasi antrian dan data metrik. Untuk informasi lebih lanjut tentang antrian, lihat [Mengelola penempatan sesi permainan dengan Amazon GameLift Servers mengantre](#)

Halaman antrian menampilkan informasi ringkasan berikut untuk setiap antrian. Anda dapat menyesuaikan konten tabel sesuai kebutuhan menggunakan alat Preferensi (lihat



ikon di sudut kanan atas tabel). Preferensi khusus disimpan ke pengguna AWS akun Anda dan diterapkan secara otomatis setiap kali Anda melihat halaman ini.

- Nama antrean — Nama yang ditetapkan ke antrean. Permintaan untuk sesi game baru menentukan antrean berdasarkan nama ini.
- Batas waktu antrean — Durasi maksimum waktu, dalam hitungan detik, bahwa permintaan penempatan sesi game tetap dalam antrean sebelum waktu habis.
- Tujuan dalam antrean — Jumlah armada yang tercantum dalam konfigurasi antrean. Amazon GameLift Servers menempatkan sesi permainan baru pada armada apa pun dalam antrian.

Lihat detail antrian

Anda dapat mengakses informasi detail tentang antrean apa pun, termasuk konfigurasi dan metrik antrean. Untuk membuka halaman detail antrian, buka halaman Antrian dan pilih nama antrian.

Halaman detail antrean menampilkan tabel ringkasan dan tab yang berisi informasi tambahan. Di halaman ini Anda dapat melakukan hal berikut:

- Memperbarui konfigurasi antrean, daftar tujuan dan kebijakan latensi pemain. Pilih Edit.
- Hapus antrian. Setelah Anda menghapus antrian, semua permintaan untuk sesi permainan baru yang mereferensikan nama antrian tersebut akan gagal. Pilih Hapus.

Note

Untuk memulihkan antrian yang dihapus, buat antrian baru dengan nama antrian yang dihapus.

Detail

Gambaran Umum

Bagian Ikhtisar menampilkan Nama Sumber Daya Amazon (ARN) antrian dan Timeout. Anda dapat menggunakan ARN saat mereferensikan antrian dalam tindakan atau area lain Amazon GameLift Servers. Batas waktu adalah durasi waktu maksimum, dalam hitungan detik, permintaan penempatan sesi permainan tetap dalam antrian sebelum waktu habis.

Pemberitahuan peristiwa

Bagian Pemberitahuan acara mencantumkan topik SNS Amazon GameLift Servers menerbitkan pemberitahuan acara ke dan data Peristiwa yang ditambahkan ke semua peristiwa yang dibuat oleh antrian ini.

Tanda

Tabel Tag menampilkan kunci dan nilai yang digunakan untuk menandai sumber daya. Untuk informasi selengkapnya tentang penandaan, lihat [AWS Menandai](#) sumber daya.

Metrik

Tab Metrik menampilkan representasi grafis metrik antrean dari waktu ke waktu.

Metrik antrian mencakup berbagai informasi yang menjelaskan aktivitas penempatan di seluruh antrian, termasuk penempatan yang berhasil diatur berdasarkan Wilayah. Anda dapat menggunakan data Wilayah untuk memahami di mana Anda menghosting game Anda. Metrik penempatan regional dapat membantu mendeteksi masalah dengan desain antrian keseluruhan.

Metrik antrian juga tersedia di Amazon. CloudWatch Untuk deskripsi metrik yang tersedia, lihat [Amazon GameLift Serversmetrik untuk antrian](#)

Tujuan

Tab Tujuan menampilkan semua armada atau alias yang terdaftar untuk antrean.

Saat Amazon GameLift Servers mencari tujuan sumber daya yang tersedia untuk menjadi tuan rumah sesi permainan baru, ia mencari urutan default yang tercantum di sini. Selama ada kapasitas pada tujuan pertama yang terdaftar, Amazon GameLift Servers menempatkan sesi permainan baru di sana. Anda dapat membiarkan permintaan penempatan sesi game individu menimpa urutan default dengan menyediakan data latensi pemain. Data ini memberi tahu Amazon GameLift Servers untuk mencari tujuan yang tersedia dengan latensi pemain rata-rata terendah. Untuk informasi selengkapnya tentang mendesain antrian Anda, lihat [Kustomisasi antrian sesi permainan](#)

Penempatan sesi

Kebijakan latensi pemain

Bagian Kebijakan latensi Pemain menampilkan semua kebijakan yang digunakan antrean. Tabel mencantumkan kebijakan dalam urutan yang diberlakukan.

Lokasi

Bagian Lokasi menunjukkan lokasi tempat antrian ini dapat menempatkan sesi permainan.

Prioritas

Bagian Prioritas menunjukkan urutan antrian mengevaluasi detail sesi permainan.

Urutan lokasi

Bagian Urutan lokasi menunjukkan urutan default yang digunakan antrian saat menempatkan sesi permainan. Antrian menggunakan urutan ini jika Anda belum menentukan jenis prioritas lainnya.

Monitor Amazon GameLift Servers dengan Amazon CloudWatch

Anda dapat memantau Amazon GameLift Servers menggunakan Amazon CloudWatch, AWS layanan yang mengumpulkan data mentah dan memprosesnya menjadi metrik yang dapat dibaca, mendekati waktu nyata. Statistik ini disimpan selama 15 bulan untuk memberikan perspektif historis tentang kinerja hosting server game Anda. Amazon GameLift Servers Anda dapat mengatur alarm yang mengawasi ambang batas tertentu dan mengirim pemberitahuan atau mengambil tindakan saat ambang batas tersebut terpenuhi. Untuk informasi selengkapnya, lihat [Panduan CloudWatch Pengguna Amazon](#).

Tabel berikut ini mencantumkan metrik dan dimensi untuk Amazon GameLift Servers. Semua metrik yang tersedia juga CloudWatch tersedia di Amazon GameLift Servers konsol, yang menyediakan

data sebagai satu set grafik yang dapat disesuaikan. Untuk mengakses CloudWatch metrik untuk game Anda AWS Management Console, gunakan AWS CLI, atau CloudWatch API.

Jika metrik tidak memiliki lokasi, ia menggunakan lokasi rumah.

Dimensi untuk metrik Amazon GameLift Servers

Amazon GameLift Servers mendukung metrik penyaringan dengan dimensi berikut.

Dimensi	Deskripsi
Location	Filter metrik untuk lokasi penyebaran armada. Jika metrik tidak memiliki lokasi, ia menggunakan lokasi rumah.
FleetId	Filter metrik untuk satu armada. Dimensi ini dapat digunakan dengan semua metrik armada untuk instans, proses server, sesi game, dan sesi pemain.
MetricGroup	Filter metrik untuk mengoleksi armada. Tambahkan armada ke grup metrik dengan menambahkan nama grup metrik ke atribut armada (lihat UpdateFleetAttributes()). Dimensi ini dapat digunakan dengan semua metrik armada untuk instans, proses server, sesi game, dan sesi pemain.
QueueName	Filter metrik untuk antrian tunggal. Dimensi ini hanya digunakan dengan metrik untuk antrian sesi game.
ConfigurationName	Filter metrik untuk konfigurasi matchmaking tunggal. Dimensi ini digunakan dengan metrik untuk konfigurasi matchmaking.
ConfigurationName-RuleName	Filter metrik untuk perpotongan konfigurasi matchmaking dan aturan matchmaking. Dimensi ini digunakan dengan metrik untuk aturan matchmaking saja.

Dimensi	Deskripsi
InstanceType	Filter metrik untuk penunjukan tipe EC2 instance, seperti "c4.large". Dimensi ini digunakan dengan metrik untuk instans spot.
OperatingSystem	Filter metrik untuk sistem operasi instans Dimensi ini digunakan dengan metrik untuk instans spot.
GameServerGroup	Filter FleetIQ metrik untuk grup server game.

Amazon GameLift Servers metrik untuk armada

Namespace `AWS/GameLift` mencakup metrik berikut yang terkait dengan aktivitas di seluruh armada atau grup armada. Armada digunakan dengan Amazon GameLift Servers solusi terkelola. Amazon GameLift Servers Layanan mengirimkan metrik ke CloudWatch setiap menit.

Instans

Metrik	Deskripsi
ActiveInstances	Instans dengan status AKTIF, yang berarti mereka menjalankan proses server secara aktif. Hitungan termasuk instans idle dan mereka yang menjadi hosting satu atau lebih sesi game. Metrik ini mengukur total kapasitas instans saat ini. Metrik ini dapat digunakan dengan penskalaan otomatis. Unit: Hitungan CloudWatch Statistik yang relevan: Rata-rata, Minimum, Maksimum Dimensi: Lokasi
DesiredInstances	Target jumlah instans aktif Amazon GameLift Servers yang bekerja untuk mempertahankan armada. Dengan penskalaan otomatis, nilai ini ditentukan

Metrik	Deskripsi
	berdasarkan kebijakan penskalaan yang saat ini berlaku. Tanpa penskalaan otomatis, nilai ini diatur secara manual. Metrik ini tidak tersedia saat melihat data untuk grup metrik armada. Unit: Hitungan CloudWatch Statistik yang relevan: Rata-rata, Minimum, Maksimum Dimensi: Lokasi
IdleInstances	Instans aktif yang saat ini hosting nol (0) sesi game. Metrik ini mengukur kapasitas yang tersedia tetapi tidak digunakan. Metrik ini dapat digunakan dengan penskalaan otomatis. Unit: Hitungan CloudWatch Statistik yang relevan: Rata-rata, Minimum, Maksimum Dimensi: Lokasi
MaxInstances	Jumlah maksimum instans yang diizinkan untuk armada. Maksimum instans armada menentukan batas kapasitas selama peningkatan manual atau penskalaan otomatis. Metrik ini tidak tersedia saat melihat data untuk grup metrik armada. Unit: Hitungan CloudWatch Statistik yang relevan: Rata-rata, Minimum, Maksimum Dimensi: Lokasi

Metrik	Deskripsi
MinInstances	Jumlah minimum instans diizinkan untuk armada. Minimum instans armada menentukan kapasitas dasar selama penskalaan otomatis atau manual. Metrik ini tidak tersedia saat melihat data untuk grup metrik armada. Unit: Hitungan CloudWatch Statistik yang relevan: Rata-rata, Minimum, Maksimum Dimensi: Lokasi
PercentIdleInstances	Persentase dari semua instans aktif yang siaga (dihitung sebagai $\text{IdleInstances} / \text{ActiveInstances}$). Metrik ini dapat digunakan untuk penskalaan otomatis. Unit: Persen CloudWatch Statistik yang relevan: Rata-rata, Minimum, Maksimum Dimensi: Lokasi
RecycledInstances	Jumlah instans spot yang telah didaur ulang dan diganti. Amazon GameLift Servers mendaur ulang instance spot yang saat ini tidak menjadi tuan rumah sesi permainan dan memiliki kemungkinan gangguan yang tinggi. Unit: Hitungan CloudWatch Statistik yang relevan: Jumlah, Rata-rata, Minimum, Maksimum Dimensi: Lokasi

Metrik	Deskripsi
InstanceInterruptions	Jumlah dari instans spot yang telah terganggu. Unit: Hitungan CloudWatch Statistik yang relevan: Jumlah, Rata-rata, Minimum, Maksimum Dimensi: Lokasi
CPUUtilization	EC2 metrik. Untuk metrik Amazon GameLift Servers ini mewakili kinerja perangkat keras di semua instans aktif di lokasi armada. Persentase waktu CPU fisik yang EC2 digunakan Amazon untuk menjalankan instance, yang mencakup waktu yang dihabiskan untuk menjalankan kode pengguna dan EC2 kode Amazon. Alat dalam sistem operasi Anda dapat menunjukkan persentase yang berbeda dari CloudWatch faktor seperti simulasi perangkat lama, konfigurasi perangkat non-warisan, beban kerja interupsi berat, migrasi langsung, dan pembaruan langsung. Unit: Persen
NetworkIn	EC2 metrik. Untuk metrik Amazon GameLift Servers ini mewakili kinerja perangkat keras di semua instans aktif di lokasi armada. Jumlah byte yang digunakan pada semua antarmuka jaringan oleh instans. Metrik ini mengidentifikasi volume lalu lintas jaringan masuk ke aplikasi pada satu instance. Unit: Byte

Metrik	Deskripsi
NetworkOut	EC2 metrik. Untuk metrik Amazon GameLift Servers ini mewakili kinerja perangkat keras di semua instans aktif di lokasi armada. Jumlah byte yang dikirimkan ke semua antarmuka jaringan oleh instans. Metrik ini mengidentifikasi volume lalu lintas jaringan keluar ke aplikasi pada satu instance. Unit: Byte
DiskReadBytes	EC2 metrik. Untuk metrik Amazon GameLift Servers ini mewakili kinerja perangkat keras di semua instans aktif di lokasi armada. Bit yang dibaca dari semua volume penyimpanan instans yang tersedia untuk instans. Metrik ini digunakan untuk menentukan volume data yang dibaca aplikasi dari hard disk instans. Anda dapat menggunakannya untuk menentukan kecepatan aplikasi. Unit: Byte
DiskWriteBytes	EC2 metrik. Untuk metrik Amazon GameLift Servers ini mewakili kinerja perangkat keras di semua instans aktif di lokasi armada. Bit yang ditulis ke semua volume penyimpanan instans yang tersedia untuk instans. Metrik ini digunakan untuk menentukan volume data yang ditulis aplikasi ke hard disk instans. Anda dapat menggunakannya untuk menentukan kecepatan aplikasi. Unit: Byte

Metrik	Deskripsi
DiskReadOps	EC2 metrik. Untuk metrik Amazon GameLift Servers ini mewakili kinerja perangkat keras di semua instans aktif di lokasi armada. Operasi baca yang diselesaikan dari semua volume penyimpanan instans yang tersedia untuk instans tersebut dalam periode waktu yang ditentukan. Untuk menghitung rata-rata I/O operasi per detik (IOPS) untuk periode tersebut, bagilah total operasi dalam periode tersebut dengan jumlah detik dalam periode tersebut. Unit: Hitungan
DiskWriteOps	EC2 metrik. Untuk metrik Amazon GameLift Servers ini mewakili kinerja perangkat keras di semua instans aktif di lokasi armada. Operasi tulis yang diselesaikan ke semua volume penyimpanan instans yang tersedia untuk instans tersebut dalam periode waktu yang ditentukan. Untuk menghitung rata-rata I/O operasi per detik (IOPS) untuk periode tersebut, bagilah total operasi dalam periode tersebut dengan jumlah detik dalam periode tersebut. Unit: Hitungan

Proses server

Metrik	Deskripsi
ActiveServerProcesses	Proses server dengan status AKTIF, yang berarti mereka berjalan dan mampu menjadi host sesi game. Penghitungan termasuk proses server siaga dan mereka yang menjadi host sesi game. Metrik ini mengukur total kapasitas proses server saat ini. Unit: Hitungan

Metrik	Deskripsi
	CloudWatch Statistik yang relevan: Rata-rata, Minimum, Maksimum Dimensi: Lokasi
HealthyServerProcesses	Proses server aktif yang melaporkan kondisi. Metrik ini berguna untuk melacak kondisi server game armada secara keseluruhan. Unit: Hitungan CloudWatch Statistik yang relevan: Rata-rata, Minimum, Maksimum Dimensi: Lokasi
PercentHealthyServerProcesses	Persentase semua proses server aktif yang melaporkan kondisi (dihitung sebagai $\text{HealthyServerProcesses} / \text{ActiveServerProcesses}$). Unit: Persen CloudWatch Statistik yang relevan: Rata-rata, Minimum, Maksimum Dimensi: Lokasi

Metrik	Deskripsi
ServerProcessAbnormalTerminations	Proses server yang ditutup karena keadaan abnormal sejak laporan terakhir. Metrik ini mencakup penghentian yang diprakarsai oleh layanan. Amazon GameLift Servers Ini terjadi ketika proses server berhenti merespons, secara konsisten melaporkan pemeriksaan kesehatan yang gagal, atau tidak berakhir dengan bersih (dengan memanggil ProcessEnding()). Unit: Hitungan CloudWatch Statistik yang relevan: Jumlah, Rata-rata, Minimum, Maksimum Dimensi: Lokasi
ServerProcessActivations	Proses server yang berhasil bertransisi dari MENGAKTIFKAN ke status AKTIF sejak laporan terakhir. Proses server tidak dapat menghosting sesi game sampai mereka aktif. Unit: Hitungan CloudWatch Statistik yang relevan: Jumlah, Rata-rata, Minimum, Maksimum Dimensi: Lokasi

Metrik	Deskripsi
ServerProcessTerminations	Proses server yang dimatikan sejak laporan terakhir. Ini mencakup semua proses server yang beralih ke status DIHENTIKAN untuk alasan apa pun, termasuk penghentian proses normal dan abnormal terminasi. Unit: Hitungan CloudWatch Statistik yang relevan: Jumlah, Rata-rata, Minimum, Maksimum Dimensi: Lokasi

Sesi game

Metrik	Deskripsi
ActivatingGameSessions	Sesi game dengan status MENGAKTIFKAN, yang berarti mereka sedang dalam proses memulai. Sesi game tidak dapat menjadi host pemain sampai mereka aktif. Angka yang tinggi untuk jangka waktu yang berkelanjutan dapat menunjukkan bahwa sesi game tidak bertransisi dari MENGAKTIFKAN ke status AKTIF. Metrik ini dapat digunakan dengan penskalaan otomatis. Unit: Hitungan CloudWatch Statistik yang relevan: Rata-rata, Minimum, Maksimum Dimensi: Lokasi
ActiveGameSessions	Sesi game dengan status AKTIF, yang berarti mereka dapat menjadi host pemain, dan menjadi host pemain nol atau lebih. Metrik ini mengukur

Metrik	Deskripsi
	jumlah total sesi game yang saat ini di-host. Metrik ini dapat digunakan dengan penskalaan otomatis. Unit: Hitungan CloudWatch Statistik yang relevan: Rata-rata, Minimum, Maksimum Dimensi: Lokasi
AvailableGameSessions	Proses server yang aktif dan sehat yang saat ini tidak digunakan untuk meng-host sesi game dan dapat memulai sesi game baru tanpa penundaan untuk menjalankan proses atau instans server baru. Metrik ini dapat digunakan dengan penskalaan otomatis.  Note Untuk armada yang membatasi aktivasi sesi game bersamaan, gunakan metrik <code>ConcurrentActivatableGameSessions</code>. Metrik tersebut lebih akurat mewakili jumlah sesi game baru yang dapat dimulai tanpa penundaan apa pun. Unit: Hitungan CloudWatch Statistik yang relevan: Rata-rata, Minimum, Maksimum Dimensi: Lokasi

Metrik	Deskripsi
<code>ConcurrentActivatableGameSessions</code>	Proses server yang aktif dan sehat yang saat ini tidak digunakan untuk meng-host sesi game dan dapat segera memulai sesi game baru. Metrik ini berbeda dari <code>AvailableGameSessions</code> dengan cara berikut: tidak menghitung proses server yang saat ini tidak dapat mengaktifkan sesi game baru karena batas pada aktivasi sesi game. (Lihat pengaturan RuntimeConfigurationOptional.armadaMaxConcurrentGameSessionActivations). Untuk armada yang tidak membatasi aktivasi sesi game, metrik ini identik dengan <code>AvailableGameSessions</code>. Unit: Hitungan CloudWatch Statistik yang relevan: Rata-rata, Minimum, Maksimum Dimensi: Lokasi
<code>PercentAvailableGameSessions</code>	Persentase slot sesi game pada semua proses server aktif (sehat atau tidak sehat) yang saat ini tidak sedang digunakan (dihitung sebagai $\frac{\text{AvailableGameSessions}}{[\text{ActiveGameSessions} + \text{AvailableGameSessions} + \text{unhealthy server processes}]}$). Metrik ini dapat digunakan dengan penskalaan otomatis. Unit: Persen CloudWatch Statistik yang relevan: Rata-rata Dimensi: Lokasi

Metrik	Deskripsi
GameSessionInterruptions	Jumlah sesi game pada instans spot yang telah terganggu. Unit: Hitungan CloudWatch Statistik yang relevan: Jumlah, Rata-rata, Minimum, Maksimum Dimensi: Lokasi

Sesi pemain

Metrik	Deskripsi
CurrentPlayerSessions	Sesi pemain dengan status AKTIF (pemain terhubung ke sesi game aktif) atau status DISIMPAN (pemain telah diberi slot dalam sesi game namun belum terhubung). Metrik ini dapat digunakan dengan penskalaan otomatis. Unit: Hitungan CloudWatch Statistik yang relevan: Rata-rata, Minimum, Maksimum
PlayerSessionActivations	Sesi pemain yang beralih dari status DISIMPAN menjadi AKTIF sejak laporan terakhir. Hal ini terjadi ketika pemain berhasil terhubung ke sesi game aktif. Unit: Hitungan CloudWatch Statistik yang relevan: Jumlah, Rata-rata, Minimum, Maksimum

Amazon GameLift Servers metrik untuk antrian

Namespace Amazon GameLift mencakup metrik berikut yang terkait dengan aktivitas di antrian penempatan sesi game. Antrian digunakan dengan solusi terkelola Amazon GameLift Servers. Amazon GameLift Servers Layanan mengirimkan metrik ke CloudWatch setiap menit.

Metrik	Deskripsi
AverageWaitTime	Jumlah rata-rata waktu bahwa permintaan penempatan sesi game dalam antrian dengan status TERTUNDA telah menunggu untuk dipenuhi. Unit: detik CloudWatch Statistik yang relevan: Rata-rata, Minimum, Maksimum, Jumlah Dimensi: Lokasi
FirstChoiceNotViable	Sesi game yang berhasil ditempatkan tetapi TIDAK dalam armada pilihan pertama, karena armada tersebut dianggap tidak layak (seperti armada spot dengan tingkat interupsi yang tinggi). Metrik ini didasarkan pada biaya, bukan latensi. Armada pilihan pertama adalah armada pertama yang terdaftar dalam antrian atau—ketika permintaan penempatan menyertakan data latensi pemain—itu adalah armada pertama yang dipilih berdasarkan prioritas. FleetIQ Jika tidak ada armada spot yang layak, setiap armada di wilayah tersebut dapat dipilih. Unit: Hitungan CloudWatch Statistik yang relevan: Rata-rata, Minimum, Maksimum, Jumlah
FirstChoiceOutOfCapacity	Sesi game yang berhasil ditempatkan tetapi TIDAK dalam armada pilihan pertama, karena armada tersebut tidak memiliki sumber daya yang tersedia.

Metrik	Deskripsi
	Armada pilihan pertama adalah armada pertama yang terdaftar dalam antrian atau—ketika permintaan penempatan menyertakan data latensi pemain—armada pertama yang dipilih berdasarkan prioritas yang Anda tentukan. FleetIQ Unit: Hitungan CloudWatch Statistik yang relevan: Rata-rata, Minimum, Maksimum, Jumlah
LowestLatencyPlacement	Sesi game yang berhasil ditempatkan di wilayah yang menawarkan latensi serendah mungkin antrian untuk para pemain. Metrik ini dipancarkan hanya bila data latensi pemain disertakan dalam permintaan penempatan. Unit: Hitungan CloudWatch Statistik yang relevan: Rata-rata, Minimum, Maksimum, Jumlah
LowestPricePlacement	Sesi game berhasil ditempatkan dalam armada dengan harga antrian serendah mungkin untuk wilayah yang dipilih. Armada ini dapat berupa armada spot atau instans sesuai permintaan jika antrian tidak memiliki instans spot. Metrik ini dipancarkan hanya bila data latensi pemain disertakan dalam permintaan penempatan. Unit: Hitungan CloudWatch Statistik yang relevan: Rata-rata, Minimum, Maksimum, Jumlah

Metrik	Deskripsi
Placement <region name>	Sesi game berhasil ditempatkan di armada yang berada di wilayah yang ditentukan. Metrik ini memecah PlacementsSucceeded metrik menurut wilayah. Unit: Hitungan CloudWatchStatistik yang relevan: Jumlah
PlacementsCanceled	Permintaan penempatan sesi game yang dibatalkan sebelum waktu habis sejak laporan terakhir. Unit: Hitungan CloudWatch Statistik yang relevan: Rata-rata, Minimum, Maksimum, Jumlah
PlacementsFailed	Permintaan penempatan sesi game yang gagal karena alasan apapun sejak laporan terakhir. Unit: Hitungan CloudWatch Statistik yang relevan: Rata-rata, Minimum, Maksimum, Jumlah
PlacementsStarted	Permintaan penempatan sesi game baru yang ditambahkan ke antrean sejak laporan terakhir. Unit: Hitungan CloudWatch Statistik yang relevan: Rata-rata, Minimum, Maksimum, Jumlah

Metrik	Deskripsi
PlacementsSucceeded	Permintaan penempatan sesi game yang menghasilkan sesi game baru sejak laporan terakhir. Unit: Hitungan CloudWatch Statistik yang relevan: Rata-rata, Minimum, Maksimum, Jumlah
PlacementsTimedOut	Permintaan penempatan sesi game yang mencapai batas waktu antrean tanpa terpenuhi sejak laporan terakhir. Unit: Hitungan CloudWatch Statistik yang relevan: Rata-rata, Minimum, Maksimum, Jumlah
QueueDepth	Jumlah permintaan penempatan sesi game dalam antrean dengan status TERTUNDA. Unit: Hitungan CloudWatch Statistik yang relevan: Rata-rata, Minimum, Maksimum, Jumlah Dimensi: Lokasi

Amazon GameLift Serversmetrik untuk perjudohan

Amazon GameLift ServersNamespace menyertakan metrik FlexMatch aktivitas untuk konfigurasi perjudohan dan aturan perjudohan. FlexMatchperjudohan digunakan dengan solusi terkelolaAmazon GameLift Servers. Amazon GameLift ServersLayanan mengirimkan metrik ke CloudWatch setiap menit.

Untuk informasi selengkapnya tentang urutan aktivitas perjudohan, lihat [Cara Amazon GameLift ServersFlexMatch](#) kerja.

Konfigurasi matchmaking

Metrik	Deskripsi
CurrentTickets	Permintaan matchmaking saat ini sedang diproses atau menunggu untuk diproses. Unit: Hitungan CloudWatch Statistik yang relevan: Rata-rata, Minimum, Maksimum, Jumlah
MatchAcceptancesTimedOut	Untuk konfigurasi matchmaking yang memerlukan penerimaan, pertandingan potensial yang habis waktunya selama penerimaan sejak laporan terakhir. Unit: Hitungan CloudWatch Statistik yang relevan: Jumlah
MatchesAccepted	Untuk konfigurasi matchmaking yang memerlukan penerimaan, pertandingan potensial yang diterima sejak laporan terakhir. Unit: Hitungan CloudWatch Statistik yang relevan: Jumlah
MatchesCreated	Potensi pertandingan yang dibuat sejak laporan terakhir. Unit: Hitungan CloudWatch Statistik yang relevan: Jumlah
MatchesPlaced	Pertandingan yang berhasil ditempatkan ke dalam sesi game sejak laporan terakhir. Unit: Hitungan CloudWatch Statistik yang relevan: Jumlah

Metrik	Deskripsi
MatchesRejected	Untuk konfigurasi matchmaking yang memerlukan penerimaan, pertandingan potensial yang ditolak oleh setidaknya satu pemain sejak laporan terakhir. Unit: Hitungan CloudWatch Statistik yang relevan: Jumlah
PlayersStarted	Pemain dalam tiket matchmaking yang ditambahkan sejak laporan terakhir. Unit: Hitungan CloudWatch Statistik yang relevan: Jumlah
TicketsFailed	Permintaan matchmaking yang mengakibatkan kegagalan sejak laporan terakhir. Unit: Hitungan CloudWatch Statistik yang relevan: Jumlah
TicketsStarted	Permintaan matchmaking baru yang dibuat sejak laporan terakhir. Unit: Hitungan CloudWatch Statistik yang relevan: Jumlah
TicketsTimedOut	Permintaan matchmaking yang mencapai batas waktu habis sejak laporan terakhir. Unit: Hitungan CloudWatch Statistik yang relevan: Jumlah

Metrik	Deskripsi
TimeToMatch	Untuk permintaan matchmaking yang dimasukkan ke dalam pertandingan potensial sebelum laporan terakhir, jumlah waktu antara pembuatan tiket dan kemungkinan pembuatan pertandingan. Unit: detik CloudWatch Statistik yang relevan: Sampel Data, Rata-rata, Minimum, Maksimum
TimeToTicketCancel	Untuk permintaan matchmaking yang dibatalkan sebelum laporan terakhir, jumlah waktu antara pembuatan tiket dan pembatalan. Unit: detik CloudWatch Statistik yang relevan: Sampel Data, Rata-rata, Minimum, Maksimum
TimeToTicketSuccess	Untuk permintaan matchmaking yang berhasil sebelum laporan terakhir, jumlah waktu antara pembuatan tiket dan penempatan pertandingan yang sukses. Unit: detik CloudWatch Statistik yang relevan: Sampel Data, Rata-rata, Minimum, Maksimum

Aturan matchmaking

Metrik	Deskripsi
RuleEvaluationsPassed	Evaluasi aturan selama proses matchmaking yang berlalu sejak laporan terakhir. Metrik ini terbatas pada 50 aturan teratas.

Metrik	Deskripsi
	Unit: Hitungan CloudWatchStatistik yang relevan: Jumlah
RuleEvaluationsFailed	Evaluasi aturan selama matchmaking yang gagal sejak laporan terakhir. Metrik ini terbatas pada 50 aturan teratas. Unit: Hitungan CloudWatch Statistik yang relevan: Jumlah

Amazon GameLift Serversmetrik untuk FleetIQ

Amazon GameLift ServersNamespace mencakup metrik untuk grup server FleetIQ game dan aktivitas server game sebagai bagian dari solusi FleetIQ mandiri untuk hosting game. Amazon GameLift ServersLayanan mengirimkan metrik ke CloudWatch setiap menit. Lihat juga [Memantau grup dan instans Auto Scaling menggunakan amazon CloudWatch di Panduan Pengguna Amazon EC2 Auto Scaling](#).

Metrik	Deskripsi
AvailableGameServers	Server game yang tersedia untuk menjalankan eksekusi game dan saat ini tidak ditempati dengan gameplay. Nomor ini mencakup server game yang telah diklaim tetapi masih dalam status tersedianya. Unit: Hitungan CloudWatch Statistik Amazon yang relevan: Jumlah Dimensi: GameServerGroup
UtilizedGameServers	Server game yang saat ini ditempati dengan gameplay. Nomor ini mencakup server game yang dalam status DIMANFAATKAN.

Metrik	Deskripsi
	Unit: Hitungan CloudWatch Statistik Amazon yang relevan: Jumlah Dimensi: GameServerGroup
DrainingAvailableGameServers	Server game pada instans yang dijadwalkan untuk dihentikan karena saat ini tidak mendukung gameplay. Server game ini adalah prioritas terendah untuk diklaim sebagai tanggapan atas permintaan klaim baru. Unit: Hitungan CloudWatch Statistik Amazon yang relevan: Jumlah Dimensi: GameServerGroup
DrainingUtilizedGameServers	Server game pada instans yang dijadwalkan untuk dihentikan yang saat ini men-support gameplay. Unit: Hitungan CloudWatch Statistik Amazon yang relevan: Jumlah Dimensi: GameServerGroup

Metrik	Deskripsi
PercentUtilizedGameServers	Porsi server game yang saat ini men-support eksekusi game. Metrik ini menunjukkan jumlah kapasitas server game yang saat ini digunakan. Hal ini berguna untuk mendorong kebijakan Auto Scaling yang dinamis dapat menambah dan menghapus instans untuk pertandingan dengan permintaan pemain. Unit: Persen CloudWatch Statistik Amazon yang relevan: Rata-rata, Minimum, Maksimum Dimensi: GameServerGroup
GameServerInterruptions	Server game di Instans spot yang terganggu karena ketersediaan Spot terbatas. Unit: Hitungan CloudWatch Statistik Amazon yang relevan: Jumlah Dimensi: GameServerGroup, InstanceType
InstanceInterruptions	Instans Spot yang terganggu karena ketersediaan yang terbatas. Unit: Hitungan CloudWatch Statistik Amazon yang relevan: Jumlah Dimensi: GameServerGroup, InstanceType

Pencatatan log Amazon GameLift Servers Panggilan API dengan AWS CloudTrail

Amazon GameLift Servers terintegrasi dengan AWS CloudTrail, layanan yang menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau AWS layanan di Amazon GameLift Servers. CloudTrail menangkap semua panggilan API untuk Amazon GameLift Servers sebagai peristiwa. Panggilan yang ditangkap termasuk panggilan dari Amazon GameLift Servers konsol dan kode panggilan ke Amazon GameLift Servers Operasi API. Jika Anda membuat jejak, Anda dapat mengaktifkan pengiriman CloudTrail acara secara berkelanjutan ke bucket Amazon S3, termasuk acara untuk Amazon GameLift Servers. Jika Anda tidak mengonfigurasi jejak, Anda masih dapat melihat peristiwa terbaru di CloudTrail konsol dalam Riwayat acara. Dengan menggunakan informasi yang dikumpulkan oleh CloudTrail, Anda dapat menentukan permintaan yang dibuat Amazon GameLift Servers, alamat IP dari mana permintaan dibuat, siapa yang membuat permintaan, kapan dibuat, dan detail tambahan.

Untuk mempelajari selengkapnya CloudTrail, lihat [Panduan AWS CloudTrail Pengguna](#).

Amazon GameLift Servers informasi di CloudTrail

CloudTrail diaktifkan pada Akun AWS saat Anda membuat akun. Ketika aktivitas terjadi di Amazon GameLift Servers, kegiatan tersebut dicatat dalam suatu CloudTrail peristiwa bersama dengan acara AWS layanan lainnya dalam riwayat Acara. Anda dapat melihat, mencari, dan mengunduh acara terbaru di situs Anda Akun AWS. Untuk informasi selengkapnya, lihat [Melihat peristiwa dengan Riwayat CloudTrail acara](#).

Untuk catatan acara yang sedang berlangsung di Anda Akun AWS, termasuk acara untuk Amazon GameLift Servers, buat jejak. Jejak memungkinkan CloudTrail untuk mengirimkan file log ke bucket Amazon S3. Secara default, saat Anda membuat jejak di konsol, jejak tersebut berlaku untuk semua Wilayah AWS. Jejak mencatat peristiwa dari semua Wilayah di AWS partisi dan mengirimkan file log ke bucket Amazon S3 yang Anda tentukan. Selain itu, Anda dapat mengonfigurasi AWS layanan lain untuk menganalisis lebih lanjut dan menindaklanjuti data peristiwa yang dikumpulkan dalam CloudTrail log. Untuk informasi selengkapnya, lihat berikut:

- [Gambaran umum untuk membuat jejak](#)
- [CloudTrail layanan dan integrasi yang didukung](#)
- [Mengonfigurasi notifikasi Amazon SNS untuk CloudTrail](#)

- [Menerima file CloudTrail log dari beberapa wilayah](#) dan [Menerima file CloudTrail log dari beberapa akun](#)

Semua Amazon GameLift Servers tindakan dicatat oleh CloudTrail dan didokumentasikan dalam [Amazon GameLift Servers Referensi API](#). Misalnya, panggilan `keCreateGameSession`, `CreatePlayerSession` dan `UpdateGameSession` tindakan menghasilkan entri dalam file CloudTrail log.

Setiap entri peristiwa atau log berisi informasi tentang siapa yang membuat permintaan tersebut. Informasi identitas membantu Anda menentukan berikut ini:

- Apakah permintaan itu dibuat dengan kredensial pengguna root atau AWS Identity and Access Management (IAM).
- Apakah permintaan tersebut dibuat dengan kredensial keamanan sementara untuk satu peran atau pengguna gabungan.
- Apakah permintaan itu dibuat oleh AWS layanan lain.

Untuk informasi selengkapnya, lihat [Elemen userIdentity CloudTrail](#).

Memahami Amazon GameLift Servers entri file log

Trail adalah konfigurasi yang memungkinkan pengiriman peristiwa sebagai file log ke bucket Amazon S3 yang Anda tentukan. CloudTrail file log berisi satu atau lebih entri log. Peristiwa mewakili permintaan tunggal dari sumber mana pun dan mencakup informasi tentang tindakan yang diminta, tanggal dan waktu tindakan, parameter permintaan, dan sebagainya. CloudTrail file log bukanlah jejak tumpukan yang diurutkan dari panggilan API publik, jadi file tersebut tidak muncul dalam urutan tertentu.

Contoh berikut menunjukkan entri CloudTrail log yang menunjukkan `CreateFleet` dan `DescribeFleetAttributes` tindakan.

```
{
  "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"
},
]
}

```

Mencatat pesan server di Amazon GameLift Servers

Anda dapat menangkap pesan server khusus dari Amazon GameLift Servers server Anda dalam file log. Cara Anda mengonfigurasi logging tergantung pada apakah Anda menggunakan server khusus atau Amazon GameLift Servers Realtime (lihat subbagian yang sesuai di bagian ini).

Topik

- [Pencatatan pesan server \(server khusus\)](#)
- [Pencatatan pesan server \(Amazon GameLift ServersRealtime\)](#)

Pencatatan pesan server (server khusus)

Anda dapat menangkap pesan server kustom dari server Amazon GameLift Servers kustom Anda dalam file log. Untuk mempelajari tentang pencatatan Amazon GameLift ServersRealtime, lihat [Pencatatan pesan server \(Amazon GameLift ServersRealtime\)](#).

Important

Ada batasan ukuran file log per sesi game (lihat [Amazon GameLift Server statistik akhir dan kuota](#) di Referensi Umum AWS). Saat sesi game berakhir, Amazon GameLift Servers unggah log server ke Amazon Simple Storage Service (Amazon S3). Amazon GameLift Server tidak akan mengunggah log yang melebihi batas. Log dapat tumbuh sangat cepat dan melebihi batas ukuran. Anda harus memantau log Anda dan membatasi output log hanya untuk pesan yang diperlukan.

Mengkonfigurasi logging untuk server kustom

Dengan server Amazon GameLift Servers kustom, Anda menulis kode Anda sendiri untuk melakukan logging, yang Anda konfigurasikan sebagai bagian dari konfigurasi proses server Anda. Amazon GameLift Servers menggunakan konfigurasi logging Anda untuk mengidentifikasi file yang harus diunggah ke Amazon S3 di akhir setiap sesi permainan.

Petunjuk berikut menunjukkan cara mengkonfigurasi logging menggunakan contoh kode yang disederhanakan:

C++

Untuk mengkonfigurasi logging (C++)

1. Buat vektor string yang merupakan jalur direktori ke file log server game.

```
std::string serverLog("serverOut.log");           // Example server log file
std::vector<std::string> logPaths;
```

```
logPaths.push_back(serverLog);
```

2. Berikan vektor Anda sebagai [LogParametersProcessParameters](#) objek Anda.

```
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. Berikan [ProcessParameters](#) objek saat Anda memanggil [ProcessReady\(\)](#).

```
Aws::GameLift::GenericOutcome outcome =
    Aws::GameLift::Server::ProcessReady(processReadyParameter);
```

Untuk contoh yang lebih lengkap, lihat [ProcessReady\(\)](#).

C#

Untuk mengkonfigurasi logging (C #)

1. Buat daftar string yang merupakan jalur direktori ke file log server game.

```
List<string> logPaths = new List<string>();
logPaths.Add("C:\\game\\serverOut.txt");    // Example of a log file that the
game server writes
```

2. Berikan daftar Anda sebagai [LogParametersProcessParameters](#) objek Anda.

```
var processReadyParameter = new ProcessParameters(
    this.OnGameSession,
    this.OnProcessTerminate,
    this.OnHealthCheck,
    this.OnGameSessionUpdate,
    port,
    new LogParameters(logPaths));
```

3. Berikan [ProcessParameters](#) objek saat Anda memanggil [ProcessReady\(\)](#).

```
var processReadyOutcome =  
    GameLiftServerAPI.ProcessReady(processReadyParameter);
```

Untuk contoh yang lebih lengkap, lihat [ProcessReady\(\)](#).

Menulis ke log

File log Anda ada setelah proses server Anda dimulai. Anda dapat menulis ke log menggunakan metode apa pun untuk menulis ke file. Untuk menangkap semua output standar dan keluaran kesalahan server Anda, petakan ulang aliran output ke file log, seperti pada contoh berikut:

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"));
```

Mengakses log server

Ketika sesi permainan berakhir, Amazon GameLift Servers secara otomatis menyimpan log di ember Amazon S3 dan menyimpannya selama 14 hari. Untuk mendapatkan lokasi log untuk sesi game, Anda dapat menggunakan operasi [GetGameSessionLogUrl](#) API. Untuk mengunduh log, gunakan URL yang dikembalikan operasi.

Pencatatan pesan server (Amazon GameLift ServersRealtime)

Anda dapat menangkap pesan server khusus dari file log Amazon GameLift Servers Realtime masuk Anda. Untuk mempelajari tentang pencatatan untuk server kustom, lihat [Pencatatan pesan server \(server khusus\)](#).

Ada berbagai jenis pesan yang dapat Anda output ke file log Anda (lihat [Mencatat pesan di skrip server Anda](#)). Selain pesan kustom Anda, pesan sistem Amazon GameLift Servers Realtime

keluaran Anda menggunakan jenis pesan yang sama dan menulis ke file log yang sama. Anda dapat menyesuaikan tingkat logging untuk armada Anda untuk mengurangi jumlah pesan logging yang dihasilkan server Anda (lihat [Menyesuaikan level logging](#)).

Important

Ada batasan ukuran file log per sesi game (lihat [Amazon GameLift Server statistik akhir dan kuota](#) di Referensi Umum AWS). Saat sesi game berakhir, Amazon GameLift Servers unggah log server ke Amazon Simple Storage Service (Amazon S3). Amazon GameLift Server tidak akan mengunggah log yang melebihi batas. Log dapat tumbuh sangat cepat dan melebihi batas ukuran. Anda harus memantau log Anda dan membatasi output log hanya untuk pesan yang diperlukan.

Mencatat pesan di skrip server Anda

Anda dapat menampilkan pesan khusus dalam [skrip untuk Anda Amazon GameLift Servers Realtime](#). Gunakan langkah-langkah berikut untuk mengirim pesan server ke file log:

1. Buat variable untuk menahan referensi ke objek logger.

```
var logger;
```

2. Dalam `init()` fungsi, dapatkan logger dari objek sesi dan tetapkan ke variabel logger Anda.

```
function init(rtSession) {  
    session = rtSession;  
    logger = session.getLogger();  
}
```

3. Panggil fungsi yang sesuai pada logger untuk mengeluarkan pesan.

Pesan debug

```
logger.debug("This is my debug message...");
```

Pesan informasi

```
logger.info("This is my info message...");
```


Pesan peringatan

```
logger.warn("This is my warn message...");
```

Pesan galat

```
logger.error("This is my error message...");
```

Pesan kesalahan fatal

```
logger.fatal("This is my fatal error message...");
```

Pelanggan mengalami pesan kesalahan fatal

```
logger.cxfatal("This is my customer experience fatal error message...");
```

Untuk contoh pernyataan logging dalam skrip, lihat [Amazon GameLift Servers Realtime contoh skrip](#).

Output dalam file log menunjukkan jenis pesan (DEBUG,,INFO,WARN, ERROR,FATAL,CXFATAL), seperti yang ditunjukkan pada baris berikut dari log contoh:

```
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.
```

Mengakses log server

Ketika sesi permainan berakhir, Amazon GameLift Servers secara otomatis menyimpan log di Amazon S3 dan menyimpannya selama 14 hari. Anda dapat menggunakan [panggilan `GetGameSessionLogUrl` API](#) untuk mendapatkan lokasi log untuk sesi game. Gunakan URL yang dikembalikan oleh panggilan API untuk mengunduh log.

Menyesuaikan level logging

Log dapat tumbuh sangat cepat dan melebihi batas ukuran. Anda harus memantau log Anda dan membatasi output log hanya untuk pesan yang diperlukan. Untuk Amazon GameLift ServersRealtime, Anda dapat menyesuaikan level logging dengan memberikan parameter dalam konfigurasi runtime armada Anda dalam formulir `loggingLevel: LOGGING_LEVEL`, di `LOGGING_LEVEL` mana salah satu nilai berikut:

1. debug
2. info(default)
3. warn
4. error
5. fatal
6. cxfatal

Daftar ini diurutkan dari yang paling parah (debug) hingga yang paling parah (cxfatal). Anda menetapkan satu `loggingLevel` dan server hanya akan mencatat pesan pada tingkat keparahan atau tingkat keparahan yang lebih tinggi. Misalnya, pengaturan `loggingLevel:error` akan membuat semua server di armada Anda hanya menulis `error`, `fatal`, dan `cxfatal` pesan ke log.

Anda dapat mengatur tingkat logging untuk armada Anda saat Anda membuatnya atau setelah itu berjalan. Mengubah level logging armada Anda setelah berjalan hanya akan memengaruhi log untuk sesi game yang dibuat setelah pembaruan. Log untuk setiap sesi game yang ada tidak akan terpengaruh. Jika Anda tidak menetapkan tingkat pencatatan saat membuat armada, server Anda akan menyetel level logging secara `info default`. Lihat bagian berikut untuk instruksi untuk mengatur tingkat logging.

Mengatur level logging saat membuat Amazon GameLift ServersRealtime armada (Konsol)

Ikuti petunjuk di [Buat EC2 armada Amazon GameLift Servers terkelola](#) untuk membuat armada Anda, dengan tambahan berikut:

- Dalam sublangkah alokasi proses Server dari langkah manajemen Proses, berikan pasangan nilai kunci tingkat pencatatan (seperti `loggingLevel:error`) sebagai nilai untuk parameter Peluncuran. Gunakan karakter non-alfanumerik (kecuali koma) untuk memisahkan level logging dari parameter tambahan (misalnya, `loggingLevel:error +map Winter444`

Mengatur level logging saat membuat Amazon GameLift ServersRealtime fleet (AWS CLI)

Ikuti petunjuk di [Buat EC2 armada Amazon GameLift Servers terkelola](#) untuk membuat armada Anda, dengan tambahan berikut:

- Dalam argumen `--runtime-configuration` parameter untuk [create-fleet](#), berikan pasangan nilai kunci tingkat logging (seperti `loggingLevel:error`) sebagai nilai untuk. `Parameters` Gunakan karakter non-alfanumerik (kecuali koma) untuk memisahkan level logging dari parameter tambahan apa pun. Lihat contoh berikut ini:

```
--runtime-configuration "GameSessionActivationTimeoutSeconds=60,
                        MaxConcurrentGameSessionActivations=2,
                        ServerProcesses=[{LaunchPath=/local/game/myRealtimeLaunchScript.js,
                        Parameters=loggingLevel:error +map Winter444,
                        ConcurrentExecutions=10}]"
```

Menyetel level logging untuk Amazon GameLift ServersRealtime armada yang sedang berjalan (Konsol)

Ikuti petunjuk di [Perbarui konfigurasi Amazon GameLift Servers armada](#) untuk memperbarui armada Anda menggunakan Amazon GameLift Servers Konsol, dengan tambahan berikut:

- Pada halaman Edit armada, di bawah alokasi proses Server, berikan pasangan nilai kunci tingkat pencatatan (seperti `loggingLevel:error`) sebagai nilai untuk parameter Peluncuran. Gunakan karakter non-alfanumerik (kecuali koma) untuk memisahkan level logging dari parameter tambahan (misalnya,). `loggingLevel:error +map Winter444`

Menyetel level logging untuk Amazon GameLift ServersRealtime armada yang sedang berjalan (AWS CLI)

Ikuti petunjuk di [Perbarui konfigurasi Amazon GameLift Servers armada](#) untuk memperbarui armada Anda menggunakan AWS CLI, dengan tambahan berikut:

- Dalam argumen `--runtime-configuration` parameter untuk [update-runtime-configuration](#), berikan pasangan nilai kunci tingkat logging (seperti `loggingLevel:error`) sebagai nilai untuk. `Parameters` Gunakan karakter non-alfanumerik (kecuali koma) untuk memisahkan level logging dari parameter tambahan apa pun. Lihat contoh berikut ini:

```
--runtime-configuration "GameSessionActivationTimeoutSeconds=60,  
    MaxConcurrentGameSessionActivations=2,  
    ServerProcesses=[{LaunchPath=/local/game/myRealtimeLaunchScript.js,  
        Parameters=loggingLevel:error +map Winter444,  
        ConcurrentExecutions=10}]"
```

Keamanan di Amazon GameLift Servers

Jika Anda menggunakan Amazon GameLift Servers FleetIQ sebagai fitur mandiri dengan Amazon EC2, lihat [Keamanan di Amazon EC2](#) di Panduan EC2 Pengguna Amazon.

Keamanan cloud di AWS adalah prioritas tertinggi. Sebagai pelanggan AWS, Anda mendapatkan manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan dari organisasi yang paling sensitif terhadap keamanan.

Keamanan adalah tanggung jawab bersama antara Anda AWS dan Anda. [Model tanggung jawab bersama](#) menjelaskan hal ini sebagai keamanan cloud dan keamanan dalam cloud:

- Keamanan cloud — AWS bertanggung jawab untuk melindungi infrastruktur yang menjalankan AWS layanan di AWS Cloud. AWS juga memberi Anda layanan yang dapat Anda gunakan dengan aman. Auditor pihak ketiga secara teratur menguji dan memverifikasi efektivitas keamanan kami sebagai bagian dari [program AWS kepatuhan program AWS](#). Untuk mempelajari tentang program kepatuhan yang berlaku Amazon GameLift Servers, lihat [AWS layanan dalam lingkup oleh AWS layanan program kepatuhan](#).
- Keamanan di cloud — Tanggung jawab Anda ditentukan oleh AWS layanan yang Anda gunakan. Anda juga bertanggung jawab atas faktor-faktor lain termasuk sensitivitas data Anda, persyaratan perusahaan Anda, dan I AWS dan peraturan yang berlaku.

Dokumentasi ini membantu Anda memahami cara menerapkan model tanggung jawab bersama saat menggunakan Amazon GameLift Servers. Topik berikut menunjukkan cara mengkonfigurasi Amazon GameLift Servers untuk memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga belajar cara menggunakan AWS layanan lain yang membantu Anda memantau dan mengamankan Amazon GameLift Servers sumber daya.

Topik

- [Perlindungan data di Amazon GameLift Servers](#)
- [Identitas dan manajemen akses untuk Amazon GameLift Servers](#)
- [Pencatatan dan pemantauan dengan Amazon GameLift Servers](#)
- [Validasi kepatuhan untuk Amazon GameLift Servers](#)
- [Ketahanan di Amazon GameLift Servers](#)
- [Keamanan infrastruktur di Amazon GameLift Servers](#)

- [Analisis konfigurasi dan kerentanan di Amazon GameLift Servers](#)
- [Praktik terbaik keamanan untuk Amazon GameLift Servers](#)

Perlindungan data di Amazon GameLift Servers

Jika Anda menggunakan Amazon GameLift Servers FleetIQ fitur mandiri dengan Amazon EC2, lihat [Keamanan di Amazon EC2](#) di Panduan EC2 Pengguna Amazon.

[Model tanggung jawab AWS bersama model](#) berlaku untuk perlindungan data di Amazon GameLift Servers. Seperti yang dijelaskan dalam model AWS ini, bertanggung jawab untuk melindungi infrastruktur global yang menjalankan semua AWS Cloud. Anda bertanggung jawab untuk mempertahankan kendali atas konten yang di-host pada infrastruktur ini. Anda juga bertanggung jawab atas tugas-tugas konfigurasi dan manajemen keamanan untuk Layanan AWS yang Anda gunakan. Lihat informasi yang lebih lengkap tentang privasi data dalam [Pertanyaan Umum Privasi Data](#). Lihat informasi tentang perlindungan data di Eropa di pos blog [Model Tanggung Jawab Bersama dan GDPR AWS](#) di Blog Keamanan AWS .

Untuk tujuan perlindungan data, kami menyarankan Anda melindungi Akun AWS kredensial dan mengatur pengguna individu dengan AWS IAM Identity Center atau AWS Identity and Access Management (IAM). Dengan cara itu, setiap pengguna hanya diberi izin yang diperlukan untuk memenuhi tanggung jawab tugasnya. Kami juga menyarankan supaya Anda mengamankan data dengan cara-cara berikut:

- Gunakan autentikasi multi-faktor (MFA) pada setiap akun.
- Gunakan SSL/TLS untuk berkomunikasi dengan AWS sumber daya. Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- Siapkan API dan pencatatan aktivitas pengguna dengan AWS CloudTrail. Untuk informasi tentang penggunaan CloudTrail jejak untuk menangkap AWS aktivitas, lihat [Bekerja dengan CloudTrail jejak](#) di AWS CloudTrail Panduan Pengguna.
- Gunakan solusi AWS enkripsi, bersama dengan semua kontrol keamanan default di dalamnya Layanan AWS.
- Gunakan layanan keamanan terkelola tingkat lanjut seperti Amazon Macie, yang membantu menemukan dan mengamankan data sensitif yang disimpan di Amazon S3.
- Jika Anda memerlukan modul kriptografi tervalidasi FIPS 140-3 saat mengakses AWS melalui antarmuka baris perintah atau API, gunakan titik akhir FIPS. Lihat informasi selengkapnya tentang titik akhir FIPS yang tersedia di [Standar Pemrosesan Informasi Federal \(FIPS\) 140-3](#).

Kami sangat merekomendasikan agar Anda tidak pernah memasukkan informasi identifikasi yang sensitif, seperti nomor rekening pelanggan Anda, ke dalam tanda atau bidang isian bebas seperti bidang Nama. Ini termasuk saat Anda bekerja dengan Amazon GameLift Servers atau lainnya Layanan AWS menggunakan konsol, API AWS CLI, atau AWS SDKs. Data apa pun yang Anda masukkan ke dalam tanda atau bidang isian bebas yang digunakan untuk nama dapat digunakan untuk log penagihan atau log diagnostik. Saat Anda memberikan URL ke server eksternal, kami sangat menganjurkan supaya Anda tidak menyertakan informasi kredensial di dalam URL untuk memvalidasi permintaan Anda ke server itu.

Amazon GameLift Servers-data spesifik ditangani sebagai berikut:

- Build server game dan skrip yang Anda unggah Amazon GameLift Servers disimpan di Amazon S3. Tidak ada akses pelanggan langsung ke data ini setelah diunggah. Pengguna yang berwenang bisa mendapatkan akses sementara untuk meng-unggah file, tetapi tidak dapat melihat atau memperbarui file di Amazon S3 langsung. Untuk menghapus skrip dan build, gunakan Amazon GameLift Servers konsol atau API layanan.
- Data log sesi game disimpan di Amazon S3 untuk jangka waktu terbatas setelah sesi game selesai. Pengguna yang berwenang dapat mengakses data log dengan mengunduhnya melalui tautan di Amazon GameLift Servers konsol atau dengan panggilan ke API layanan.
- Data metrik dan peristiwa disimpan Amazon GameLift Servers dan dapat diakses melalui Amazon GameLift Servers konsol atau dengan panggilan ke API layanan. Data dapat diambil pada armada, contoh, penempatan sesi game, tiket matchmaking, sesi game, dan sesi pemain. Data juga dapat diakses melalui Amazon CloudWatch dan CloudWatch Acara.
- Data yang disediakan pelanggan disimpan di Amazon GameLift Servers Pengguna yang berwenang dapat mengaksesnya dengan panggilan ke API layanan. Data yang berpotensi sensitif dapat mencakup data pemain, sesi pemain, dan data sesi game (termasuk info koneksi), data matchmaker, dan sebagainya.

 Note

Jika Anda memberikan pemutar khusus IDs dalam permintaan Anda, diharapkan nilai-nilai ini dianonimkan UUIDs dan tidak mengandung informasi pemain pengenalan.

Untuk informasi selengkapnya tentang perlindungan data, lihat postingan blog [model tanggung jawab bersama AWS dan Peraturan Perlindungan Data Umum \(GDPR\)](#) di Blog Keamanan AWS .

Enkripsi saat tidak digunakan

Enkripsi AT-rest dari data Amazon GameLift Servers -spesifik ditangani sebagai berikut:

- Pembuatan dan skrip server game disimpan di bucket Amazon S3 dengan enkripsi sisi server.
- Data yang disediakan pelanggan disimpan Amazon GameLift Servers dalam format terenkripsi.

Enkripsi bergerak

Koneksi ke Amazon GameLift Servers APIs dibuat melalui koneksi aman (SSL) dan diautentikasi menggunakan [AWS Signature Versi 4](#) (saat menghubungkan melalui AWS CLI atau AWS SDK, penandatanganan ditangani secara otomatis). Autentikasi dikelola menggunakan kebijakan akses IAM didefinisikan untuk kredensial keamanan yang digunakan untuk membuat sambungan.

Komunikasi langsung antara klien game dan server game adalah sebagai berikut:

- Untuk server game khusus yang di-host pada Amazon GameLift Servers sumber daya, komunikasi tidak melibatkan Amazon GameLift Servers layanan. Enkripsi komunikasi ini merupakan tanggung jawab pelanggan. Anda dapat menggunakan armada TLS-enabled untuk meminta klien game Anda mengotentikasi server game pada koneksi dan untuk mengenkripsi semua komunikasi antara klien game Anda dan server game.
- Karena Amazon GameLift Servers Realtime dengan mengaktifkan pembuatan sertifikat TLS, lalu lintas antara klien game dan Realtime server yang menggunakan SDK klien untuk Realtime dienkripsi dalam penerbangan. Lalu lintas TCP dienkripsi menggunakan TLS 1.2, dan lalu lintas UDP dienkripsi menggunakan DTLS 1.2.

Privasi lalu lintas antarjaringan

Anda dapat mengakses Amazon GameLift Servers instans Anda dari jarak jauh dengan aman. SSHRDP menyediakan saluran komunikasi yang aman untuk akses jarak jauh ke instans LinuxWindows Anda. Untuk instans yang menjalankan Windows, gunakan klien protokol desktop jauh (RDP). Dengan Amazon GameLift ServersFleetIQ, akses jarak jauh ke instans Anda menggunakan AWS Systems Manager Session Manager dan Run Command dienkripsi menggunakan TLS 1.2, dan permintaan untuk membuat koneksi ditandatangani menggunakan SiGv4. Untuk bantuan terkait menghubungkan ke Amazon GameLift Servers instance terkelola, lihat [Terhubung dari jarak jauh ke Amazon GameLift Servers contoh armada](#).

Identitas dan manajemen akses untuk Amazon GameLift Servers

AWS Identity and Access Management (IAM) adalah Layanan AWS yang membantu administrator mengontrol akses ke AWS sumber daya dengan aman. Administrator IAM mengontrol siapa yang dapat diautentikasi (masuk) dan diotorisasi (memiliki izin) untuk digunakan Amazon GameLift Servers sumber daya. IAM adalah Layanan AWS yang dapat Anda gunakan tanpa biaya tambahan.

Topik

- [Audiens](#)
- [Mengautentikasi dengan identitas](#)
- [Mengelola akses menggunakan kebijakan](#)
- [Bagaimana Amazon GameLift Servers bekerja dengan IAM](#)
- [Contoh kebijakan berbasis identitas untuk Amazon GameLift Servers](#)
- [Pemecahan Masalah Amazon GameLift Servers identitas dan akses](#)
- [AWS kebijakan terkelola untuk Amazon GameLift Servers](#)

Audiens

Bagaimana Anda menggunakan AWS Identity and Access Management (IAM) berbeda, tergantung pada pekerjaan yang Anda lakukan Amazon GameLift Servers.

Pengguna layanan — Jika Anda menggunakan Amazon GameLift Servers layanan untuk melakukan pekerjaan Anda, maka administrator Anda memberi Anda kredensi dan izin yang Anda butuhkan. Saat Anda menggunakan lebih banyak Amazon GameLift Servers fitur untuk melakukan pekerjaan Anda, Anda mungkin memerlukan izin tambahan. Memahami cara akses dikelola dapat membantu Anda meminta izin yang tepat dari administrator Anda. Jika Anda tidak dapat mengakses fitur di Amazon GameLift Servers, lihat [Pemecahan Masalah Amazon GameLift Servers identitas dan akses](#).

Administrator layanan — Jika Anda bertanggung jawab Amazon GameLift Servers sumber daya di perusahaan Anda, Anda mungkin memiliki akses penuh Amazon GameLift Servers. Tugas Anda adalah menentukan yang mana Amazon GameLift Servers fitur dan sumber daya yang harus diakses pengguna layanan Anda. Kemudian, Anda harus mengirimkan permintaan kepada administrator IAM untuk mengubah izin pengguna layanan Anda. Tinjau informasi di halaman ini untuk memahami konsep dasar IAM. Untuk mempelajari lebih lanjut tentang bagaimana perusahaan Anda dapat menggunakan IAM Amazon GameLift Servers, lihat [Bagaimana Amazon GameLift Servers bekerja dengan IAM](#).

Administrator IAM - Jika Anda seorang administrator IAM, Anda mungkin ingin mempelajari detail tentang cara menulis kebijakan untuk mengelola akses Amazon GameLift Servers. Untuk melihat contoh Amazon GameLift Servers Kebijakan berbasis identitas yang dapat Anda gunakan di IAM, lihat [Contoh kebijakan berbasis identitas untuk Amazon GameLift Servers](#)

Mengautentikasi dengan identitas

Otentikasi adalah cara Anda masuk AWS menggunakan kredensi identitas Anda. Anda harus diautentikasi (masuk ke AWS) sebagai Pengguna root akun AWS, sebagai pengguna IAM, atau dengan mengasumsikan peran IAM.

Anda dapat masuk AWS sebagai identitas federasi dengan menggunakan kredensi yang disediakan melalui sumber identitas. AWS IAM Identity Center Pengguna (IAM Identity Center), autentikasi masuk tunggal perusahaan Anda, dan kredensi Google atau Facebook Anda adalah contoh identitas federasi. Saat Anda masuk sebagai identitas terfederasi, administrator Anda sebelumnya menyiapkan federasi identitas menggunakan peran IAM. Ketika Anda mengakses AWS dengan menggunakan federasi, Anda secara tidak langsung mengambil peran.

Bergantung pada jenis pengguna Anda, Anda dapat masuk ke AWS Management Console atau portal AWS akses. Untuk informasi selengkapnya tentang masuk AWS, lihat [Cara masuk ke Panduan AWS Sign-In Pengguna Anda Akun AWS](#).

Jika Anda mengakses AWS secara terprogram, AWS sediakan kit pengembangan perangkat lunak (SDK) dan antarmuka baris perintah (CLI) untuk menandatangani permintaan Anda secara kriptografis dengan menggunakan kredensial Anda. Jika Anda tidak menggunakan AWS alat, Anda harus menandatangani permintaan sendiri. Guna mengetahui informasi selengkapnya tentang penggunaan metode yang disarankan untuk menandatangani permintaan sendiri, lihat [AWS Signature Version 4 untuk permintaan API](#) dalam Panduan Pengguna IAM.

Apa pun metode autentikasi yang digunakan, Anda mungkin diminta untuk menyediakan informasi keamanan tambahan. Misalnya, AWS merekomendasikan agar Anda menggunakan otentikasi multi-faktor (MFA) untuk meningkatkan keamanan akun Anda. Untuk mempelajari selengkapnya, lihat [Autentikasi multi-faktor](#) dalam Panduan Pengguna AWS IAM Identity Center dan [Autentikasi multi-faktor AWS di IAM](#) dalam Panduan Pengguna IAM.

Akun AWS pengguna root

Saat Anda membuat Akun AWS, Anda mulai dengan satu identitas masuk yang memiliki akses lengkap ke semua Layanan AWS dan sumber daya di akun. Identitas ini disebut pengguna Akun AWS root dan diakses dengan masuk dengan alamat email dan kata sandi yang Anda gunakan

untuk membuat akun. Kami sangat menyarankan agar Anda tidak menggunakan pengguna root untuk tugas sehari-hari. Lindungi kredensial pengguna root Anda dan gunakan kredensial tersebut untuk melakukan tugas yang hanya dapat dilakukan pengguna root. Untuk daftar lengkap tugas yang mengharuskan Anda masuk sebagai pengguna root, lihat [Tugas yang memerlukan kredensial pengguna root](#) dalam Panduan Pengguna IAM.

Identitas gabungan

Sebagai praktik terbaik, mewajibkan pengguna manusia, termasuk pengguna yang memerlukan akses administrator, untuk menggunakan federasi dengan penyedia identitas untuk mengakses Layanan AWS dengan menggunakan kredensi sementara.

Identitas federasi adalah pengguna dari direktori pengguna perusahaan Anda, penyedia identitas web, direktori Pusat Identitas AWS Directory Service, atau pengguna mana pun yang mengakses Layanan AWS dengan menggunakan kredensial yang disediakan melalui sumber identitas. Ketika identitas federasi mengakses Akun AWS, mereka mengambil peran, dan peran memberikan kredensi sementara.

Untuk manajemen akses terpusat, kami sarankan Anda menggunakan AWS IAM Identity Center. Anda dapat membuat pengguna dan grup di Pusat Identitas IAM, atau Anda dapat menghubungkan dan menyinkronkan ke sekumpulan pengguna dan grup di sumber identitas Anda sendiri untuk digunakan di semua aplikasi Akun AWS dan aplikasi Anda. Untuk informasi tentang Pusat Identitas IAM, lihat [Apakah itu Pusat Identitas IAM?](#) dalam Panduan Pengguna AWS IAM Identity Center .

Pengguna dan grup IAM

[Pengguna IAM](#) adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus untuk satu orang atau aplikasi. Jika memungkinkan, kami merekomendasikan untuk mengandalkan kredensial sementara, bukan membuat pengguna IAM yang memiliki kredensial jangka panjang seperti kata sandi dan kunci akses. Namun, jika Anda memiliki kasus penggunaan tertentu yang memerlukan kredensial jangka panjang dengan pengguna IAM, kami merekomendasikan Anda merotasi kunci akses. Untuk informasi selengkapnya, lihat [Merotasi kunci akses secara teratur untuk kasus penggunaan yang memerlukan kredensial jangka panjang](#) dalam Panduan Pengguna IAM.

[Grup IAM](#) adalah identitas yang menentukan sekumpulan pengguna IAM. Anda tidak dapat masuk sebagai grup. Anda dapat menggunakan grup untuk menentukan izin bagi beberapa pengguna sekaligus. Grup mempermudah manajemen izin untuk sejumlah besar pengguna sekaligus. Misalnya, Anda dapat meminta kelompok untuk menyebutkan IAMAdmins dan memberikan izin kepada grup tersebut untuk mengelola sumber daya IAM.

Pengguna berbeda dari peran. Pengguna secara unik terkait dengan satu orang atau aplikasi, tetapi peran dimaksudkan untuk dapat digunakan oleh siapa pun yang membutuhkannya. Pengguna memiliki kredensial jangka panjang permanen, tetapi peran memberikan kredensial sementara. Untuk mempelajari selengkapnya, lihat [Kasus penggunaan untuk pengguna IAM](#) dalam Panduan Pengguna IAM.

Peran IAM

[Peran IAM](#) adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus. Peran ini mirip dengan pengguna IAM, tetapi tidak terkait dengan orang tertentu. Untuk mengambil peran IAM sementara AWS Management Console, Anda dapat [beralih dari pengguna ke peran IAM \(konsol\)](#). Anda dapat mengambil peran dengan memanggil operasi AWS CLI atau AWS API atau dengan menggunakan URL kustom. Untuk informasi selengkapnya tentang cara menggunakan peran, lihat [Metode untuk mengambil peran](#) dalam Panduan Pengguna IAM.

Peran IAM dengan kredensial sementara berguna dalam situasi berikut:

- Akses pengguna terfederasi – Untuk menetapkan izin ke identitas terfederasi, Anda membuat peran dan menentukan izin untuk peran tersebut. Ketika identitas terfederasi mengautentikasi, identitas tersebut terhubung dengan peran dan diberi izin yang ditentukan oleh peran. Untuk informasi tentang peran untuk federasi, lihat [Buat peran untuk penyedia identitas pihak ketiga](#) dalam Panduan Pengguna IAM. Jika menggunakan Pusat Identitas IAM, Anda harus mengonfigurasi set izin. Untuk mengontrol apa yang dapat diakses identitas Anda setelah identitas tersebut diautentikasi, Pusat Identitas IAM akan mengorelasikan set izin ke peran dalam IAM. Untuk informasi tentang set izin, lihat [Set izin](#) dalam Panduan Pengguna AWS IAM Identity Center .
- Izin pengguna IAM sementara – Pengguna atau peran IAM dapat mengambil peran IAM guna mendapatkan berbagai izin secara sementara untuk tugas tertentu.
- Akses lintas akun – Anda dapat menggunakan peran IAM untuk mengizinkan seseorang (prinsipal tepercaya) di akun lain untuk mengakses sumber daya di akun Anda. Peran adalah cara utama untuk memberikan akses lintas akun. Namun, dengan beberapa Layanan AWS, Anda dapat melampirkan kebijakan langsung ke sumber daya (alih-alih menggunakan peran sebagai proxy). Untuk mempelajari perbedaan antara peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat [Akses sumber daya lintas akun di IAM](#) dalam Panduan Pengguna IAM.
- Akses lintas layanan — Beberapa Layanan AWS menggunakan fitur lain Layanan AWS. Misalnya, saat Anda melakukan panggilan dalam suatu layanan, biasanya layanan tersebut menjalankan aplikasi di Amazon EC2 atau menyimpan objek di Amazon S3. Sebuah layanan mungkin

melakukannya menggunakan izin prinsipal yang memanggil, menggunakan peran layanan, atau peran terkait layanan.

- Sesi akses teruskan (FAS) — Saat Anda menggunakan pengguna IAM atau peran untuk melakukan tindakan AWS, Anda dianggap sebagai prinsipal. Ketika Anda menggunakan beberapa layanan, Anda mungkin melakukan sebuah tindakan yang kemudian menginisiasi tindakan lain di layanan yang berbeda. FAS menggunakan izin dari pemanggilan utama Layanan AWS, dikombinasikan dengan permintaan Layanan AWS untuk membuat permintaan ke layanan hilir. Permintaan FAS hanya dibuat ketika layanan menerima permintaan yang memerlukan interaksi dengan orang lain Layanan AWS atau sumber daya untuk menyelesaikannya. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan ketika mengajukan permintaan FAS, lihat [Sesi akses maju](#).
- Peran layanan – Peran layanan adalah [peran IAM](#) yang dijalankan oleh layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, mengubah, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat [Buat sebuah peran untuk mendelegasikan izin ke Layanan AWS](#) dalam Panduan pengguna IAM.
- Peran terkait layanan — Peran terkait layanan adalah jenis peran layanan yang ditautkan ke. Layanan AWS Layanan tersebut dapat menjalankan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan muncul di Anda Akun AWS dan dimiliki oleh layanan. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran terkait layanan.
- Aplikasi yang berjalan di Amazon EC2 — Anda dapat menggunakan peran IAM untuk mengelola kredensi sementara untuk aplikasi yang berjalan pada EC2 instance dan membuat AWS CLI atau AWS permintaan API. Ini lebih baik untuk menyimpan kunci akses dalam EC2 instance. Untuk menetapkan AWS peran ke EC2 instance dan membuatnya tersedia untuk semua aplikasinya, Anda membuat profil instance yang dilampirkan ke instance. Profil instance berisi peran dan memungkinkan program yang berjalan pada EC2 instance untuk mendapatkan kredensi sementara. Untuk informasi selengkapnya, lihat [Menggunakan peran IAM untuk memberikan izin ke aplikasi yang berjalan di EC2 instans Amazon di Panduan Pengguna IAM](#).

Mengelola akses menggunakan kebijakan

Anda mengontrol akses AWS dengan membuat kebijakan dan melampirkannya ke AWS identitas atau sumber daya. Kebijakan adalah objek AWS yang, ketika dikaitkan dengan identitas atau sumber daya, menentukan izinnya. AWS mengevaluasi kebijakan ini ketika prinsipal (pengguna, pengguna root, atau sesi peran) membuat permintaan. Izin dalam kebijakan menentukan apakah permintaan diizinkan atau ditolak. Sebagian besar kebijakan disimpan AWS sebagai dokumen JSON. Untuk

informasi selengkapnya tentang struktur dan isi dokumen kebijakan JSON, lihat [Gambaran umum kebijakan JSON](#) dalam Panduan Pengguna IAM.

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Yaitu, principal dapat melakukan tindakan pada suatu sumber daya, dan dalam suatu syarat.

Secara default, pengguna dan peran tidak memiliki izin. Untuk memberikan izin kepada pengguna untuk melakukan tindakan di sumber daya yang mereka perlukan, administrator IAM dapat membuat kebijakan IAM. Administrator kemudian dapat menambahkan kebijakan IAM ke peran, dan pengguna dapat mengambil peran.

Kebijakan IAM mendefinisikan izin untuk suatu tindakan terlepas dari metode yang Anda gunakan untuk melakukan operasinya. Misalnya, anggaplah Anda memiliki kebijakan yang mengizinkan tindakan `iam:GetRole`. Pengguna dengan kebijakan tersebut bisa mendapatkan informasi peran dari AWS Management Console, API AWS CLI, atau AWS API.

Kebijakan berbasis identitas

Kebijakan berbasis identitas adalah dokumen kebijakan izin JSON yang dapat Anda lampirkan ke sebuah identitas, seperti pengguna IAM, grup pengguna IAM, atau peran IAM. Kebijakan ini mengontrol jenis tindakan yang dapat dilakukan oleh pengguna dan peran, di sumber daya mana, dan berdasarkan kondisi seperti apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat [Tentukan izin IAM kustom dengan kebijakan terkelola pelanggan](#) dalam Panduan Pengguna IAM.

Kebijakan berbasis identitas dapat dikategorikan lebih lanjut sebagai kebijakan inline atau kebijakan yang dikelola. Kebijakan inline disematkan langsung ke satu pengguna, grup, atau peran. Kebijakan terkelola adalah kebijakan mandiri yang dapat dilampirkan ke beberapa pengguna, grup, dan peran dalam. Akun AWS Kebijakan AWS terkelola mencakup kebijakan terkelola dan kebijakan yang dikelola pelanggan. Untuk mempelajari cara memilih antara kebijakan yang dikelola atau kebijakan inline, lihat [Pilih antara kebijakan yang dikelola dan kebijakan inline](#) dalam Panduan Pengguna IAM.

Kebijakan berbasis sumber daya

Kebijakan berbasis sumber daya adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya. Contoh kebijakan berbasis sumber daya adalah kebijakan kepercayaan peran IAM dan kebijakan bucket Amazon S3. Dalam layanan yang mendukung kebijakan berbasis sumber daya, administrator layanan dapat menggunakannya untuk mengontrol akses ke sumber daya tertentu.

Untuk sumber daya tempat kebijakan dilampirkan, kebijakan menentukan tindakan apa yang dapat dilakukan oleh prinsipal tertentu pada sumber daya tersebut dan dalam kondisi apa. Anda harus [menentukan prinsipal](#) dalam kebijakan berbasis sumber daya. Prinsipal dapat mencakup akun, pengguna, peran, pengguna federasi, atau. Layanan AWS

Kebijakan berbasis sumber daya merupakan kebijakan inline yang terletak di layanan tersebut. Anda tidak dapat menggunakan kebijakan AWS terkelola dari IAM dalam kebijakan berbasis sumber daya.

Daftar kontrol akses (ACLs)

Access control lists (ACLs) mengontrol prinsipal mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACLs mirip dengan kebijakan berbasis sumber daya, meskipun mereka tidak menggunakan format dokumen kebijakan JSON.

Amazon S3, AWS WAF, dan Amazon VPC adalah contoh layanan yang mendukung. ACLs Untuk mempelajari selengkapnya ACLs, lihat [Ringkasan daftar kontrol akses \(ACL\)](#) di Panduan Pengembang Layanan Penyimpanan Sederhana Amazon.

Jenis-jenis kebijakan lain

AWS mendukung jenis kebijakan tambahan yang kurang umum. Jenis-jenis kebijakan ini dapat mengatur izin maksimum yang diberikan kepada Anda oleh jenis kebijakan yang lebih umum.

- Batasan izin – Batasan izin adalah fitur lanjutan tempat Anda mengatur izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas ke entitas IAM (pengguna IAM atau peran IAM). Anda dapat menetapkan batasan izin untuk suatu entitas. Izin yang dihasilkan adalah perpotongan antara kebijakan berbasis identitas milik entitas dan batasan izinnya. Kebijakan berbasis sumber daya yang menentukan pengguna atau peran dalam bidang `Principal` tidak dibatasi oleh batasan izin. Penolakan eksplisit dalam salah satu kebijakan ini akan menggantikan pemberian izin. Untuk informasi selengkapnya tentang batasan izin, lihat [Batasan izin untuk entitas IAM](#) dalam Panduan Pengguna IAM.
- Kebijakan kontrol layanan (SCPs) — SCPs adalah kebijakan JSON yang menentukan izin maksimum untuk organisasi atau unit organisasi (OU) di. AWS Organizations AWS Organizations adalah layanan untuk mengelompokkan dan mengelola secara terpusat beberapa Akun AWS yang dimiliki bisnis Anda. Jika Anda mengaktifkan semua fitur dalam organisasi, Anda dapat menerapkan kebijakan kontrol layanan (SCPs) ke salah satu atau semua akun Anda. SCP membatasi izin untuk entitas di akun anggota, termasuk masing-masing. Pengguna root akun AWS Untuk informasi selengkapnya tentang Organizations dan SCPs, lihat [Kebijakan kontrol layanan](#) di Panduan AWS Organizations Pengguna.

- Kebijakan kontrol sumber daya (RCPs) — RCPs adalah kebijakan JSON yang dapat Anda gunakan untuk menetapkan izin maksimum yang tersedia untuk sumber daya di akun Anda tanpa memperbarui kebijakan IAM yang dilampirkan ke setiap sumber daya yang Anda miliki. RCP membatasi izin untuk sumber daya di akun anggota dan dapat memengaruhi izin efektif untuk identitas, termasuk Pengguna root akun AWS, terlepas dari apakah itu milik organisasi Anda. Untuk informasi selengkapnya tentang Organizations dan RCPs, termasuk daftar dukungan Layanan AWS tersebut RCPs, lihat [Kebijakan kontrol sumber daya \(RCPs\)](#) di Panduan AWS Organizations Pengguna.
- Kebijakan sesi – Kebijakan sesi adalah kebijakan lanjutan yang Anda berikan sebagai parameter ketika Anda membuat sesi sementara secara programatis untuk peran atau pengguna terfederasi. Izin sesi yang dihasilkan adalah perpotongan antara kebijakan berbasis identitas pengguna atau peran dan kebijakan sesi. Izin juga bisa datang dari kebijakan berbasis sumber daya. Penolakan eksplisit dalam salah satu kebijakan ini akan menggantikan pemberian izin. Untuk informasi selengkapnya, lihat [Kebijakan sesi](#) dalam Panduan Pengguna IAM.

Berbagai jenis kebijakan

Ketika beberapa jenis kebijakan berlaku pada suatu permintaan, izin yang dihasilkan lebih rumit untuk dipahami. Untuk mempelajari cara AWS menentukan apakah akan mengizinkan permintaan saat beberapa jenis kebijakan terlibat, lihat [Logika evaluasi kebijakan](#) di Panduan Pengguna IAM.

Bagaimana Amazon GameLift Servers bekerja dengan IAM

Sebelum Anda menggunakan IAM untuk mengelola akses ke Amazon GameLift Servers, pelajari fitur IAM apa yang tersedia untuk digunakan Amazon GameLift Servers.

Fitur IAM yang dapat Anda gunakan Amazon GameLift Servers

Fitur IAM	Amazon GameLift Servers dukungan
Kebijakan berbasis identitas	Ya
Kebijakan berbasis sumber daya	Tidak
Tindakan kebijakan	Ya
Sumber daya kebijakan	Ya

Fitur IAM	Amazon GameLift Servers dukungan
kunci-kunci persyaratan kebijakan (spesifik layanan)	Ya
ACLs	Tidak
ABAC (tanda dalam kebijakan)	Ya
Kredensial sementara	Ya
Izin principal	Ya
Peran layanan	Ya
Peran terkait layanan	Tidak

Untuk mendapatkan pandangan tingkat tinggi tentang bagaimana Amazon GameLift Servers dan AWS layanan lainnya bekerja dengan sebagian besar fitur IAM, lihat [AWS layanan yang bekerja dengan IAM](#) di Panduan Pengguna IAM.

Kebijakan berbasis identitas untuk Amazon GameLift Servers

Mendukung kebijakan berbasis identitas: Ya

Kebijakan berbasis identitas adalah dokumen kebijakan izin JSON yang dapat Anda lampirkan ke sebuah identitas, seperti pengguna IAM, grup pengguna IAM, atau peran IAM. Kebijakan ini mengontrol jenis tindakan yang dapat dilakukan oleh pengguna dan peran, di sumber daya mana, dan berdasarkan kondisi seperti apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat [Tentukan izin IAM kustom dengan kebijakan terkelola pelanggan](#) dalam Panduan Pengguna IAM.

Dengan kebijakan berbasis identitas IAM, Anda dapat menentukan secara spesifik apakah tindakan dan sumber daya diizinkan atau ditolak, serta kondisi yang menjadi dasar dikabulkan atau ditolaknya tindakan tersebut. Anda tidak dapat menentukan secara spesifik prinsipal dalam sebuah kebijakan berbasis identitas karena prinsipal berlaku bagi pengguna atau peran yang melekat kepadanya. Untuk mempelajari semua elemen yang dapat Anda gunakan dalam kebijakan JSON, lihat [Referensi elemen kebijakan JSON IAM](#) dalam Panduan Pengguna IAM.

Contoh kebijakan berbasis identitas untuk Amazon GameLift Servers

Untuk melihat contoh Amazon GameLift Servers Kebijakan berbasis identitas, lihat. [Contoh kebijakan berbasis identitas untuk Amazon GameLift Servers](#)

Kebijakan berbasis sumber daya dalam Amazon GameLift Servers

Mendukung kebijakan berbasis sumber daya: Tidak

Kebijakan berbasis sumber daya adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya. Contoh kebijakan berbasis sumber daya adalah kebijakan kepercayaan peran IAM dan kebijakan bucket Amazon S3. Dalam layanan yang mendukung kebijakan berbasis sumber daya, administrator layanan dapat menggunakannya untuk mengontrol akses ke sumber daya tertentu. Untuk sumber daya tempat kebijakan dilampirkan, kebijakan menentukan tindakan apa yang dapat dilakukan oleh prinsipal tertentu pada sumber daya tersebut dan dalam kondisi apa. Anda harus [menentukan prinsipal](#) dalam kebijakan berbasis sumber daya. Prinsipal dapat mencakup akun, pengguna, peran, pengguna federasi, atau. Layanan AWS

Untuk mengaktifkan akses lintas akun, Anda dapat menentukan secara spesifik seluruh akun atau entitas IAM di akun lain sebagai prinsipal dalam kebijakan berbasis sumber daya. Menambahkan prinsipal akun silang ke kebijakan berbasis sumber daya hanya setengah dari membangun hubungan kepercayaan. Ketika prinsipal dan sumber daya berbeda Akun AWS, administrator IAM di akun tepercaya juga harus memberikan izin entitas utama (pengguna atau peran) untuk mengakses sumber daya. Mereka memberikan izin dengan melampirkan kebijakan berbasis identitas kepada entitas. Namun, jika kebijakan berbasis sumber daya memberikan akses ke principal dalam akun yang sama, tidak diperlukan kebijakan berbasis identitas tambahan. Untuk informasi selengkapnya, lihat [Akses sumber daya lintas akun di IAM](#) dalam Panduan Pengguna IAM.

Tindakan kebijakan untuk Amazon GameLift Servers

Mendukung tindakan kebijakan: Ya

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Yaitu, di mana utama dapat melakukan tindakan pada sumber daya, dan dalam kondisi apa.

Elemen `Action` dari kebijakan JSON menjelaskan tindakan yang dapat Anda gunakan untuk mengizinkan atau menolak akses dalam sebuah kebijakan. Tindakan kebijakan biasanya memiliki nama yang sama dengan operasi AWS API terkait. Ada beberapa pengecualian, misalnya tindakan

hanya izin yang tidak memiliki operasi API yang cocok. Ada juga beberapa operasi yang memerlukan beberapa tindakan dalam suatu kebijakan. Tindakan tambahan ini disebut tindakan dependen.

Sertakan tindakan dalam kebijakan untuk memberikan izin untuk melakukan operasi terkait.

Untuk daftar Amazon GameLift Servers tindakan, lihat [Tindakan yang ditentukan oleh Amazon GameLift Servers](#) dalam Referensi Otorisasi Layanan.

Tindakan kebijakan di Amazon GameLift Servers gunakan awalan berikut sebelum tindakan:

```
gamelift
```

Untuk menetapkan secara spesifik beberapa tindakan dalam satu pernyataan, pisahkan tindakan tersebut dengan koma.

```
"Action": [  
    "gamelift:action1",  
    "gamelift:action2"  
]
```

Anda juga dapat menentukan beberapa tindakan menggunakan wildcard (*). Sebagai contoh, untuk menentukan semua tindakan yang dimulai dengan kata Describe, sertakan tindakan berikut:

```
"Action": "gamelift:Describe*"
```

Untuk melihat contoh Amazon GameLift Servers Kebijakan berbasis identitas, lihat. [Contoh kebijakan berbasis identitas untuk Amazon GameLift Servers](#)

Sumber daya kebijakan untuk Amazon GameLift Servers

Mendukung sumber daya kebijakan: Ya

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Yaitu, di mana utama dapat melakukan tindakan pada sumber daya, dan dalam kondisi apa.

Elemen kebijakan JSON Resource menentukan objek yang menjadi target penerapan tindakan. Pernyataan harus menyertakan elemen Resource atau NotResource. Praktik terbaiknya, tentukan sumber daya menggunakan [Amazon Resource Name \(ARN\)](#). Anda dapat melakukan ini untuk

tindakan yang mendukung jenis sumber daya tertentu, yang dikenal sebagai izin tingkat sumber daya.

Untuk tindakan yang tidak mendukung izin di tingkat sumber daya, misalnya operasi pencantuman, gunakan wildcard (*) untuk menunjukkan bahwa pernyataan tersebut berlaku untuk semua sumber daya.

```
"Resource": ""
```

Untuk daftar Amazon GameLift Servers jenis sumber daya dan mereka ARNs, lihat [Sumber daya yang ditentukan oleh Amazon GameLift Servers](#) dalam Referensi Otorisasi Layanan. Untuk mempelajari tindakan mana yang dapat Anda tentukan ARN dari setiap sumber daya, lihat [Tindakan yang ditentukan oleh Amazon GameLift Servers](#).

Beberapa Amazon GameLift Servers sumber daya memiliki nilai ARN, yang memungkinkan sumber daya untuk memiliki akses mereka dikelola menggunakan kebijakan IAM. Bagian Amazon GameLift Servers sumber daya armada memiliki ARN dengan sintaks berikut:

```
arn:${Partition}:gamelift:${Region}:${Account}:fleet/${FleetId}
```

Untuk informasi selengkapnya tentang format ARNs, lihat [Amazon Resource Names \(ARNs\)](#) di Referensi Umum AWS.

Misalnya, untuk menentukan armada `fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa` dalam pernyataan Anda, gunakan ARN berikut:

```
"Resource": "arn:aws:gamelift:us-west-2:123456789012:fleet/fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa"
```

Untuk menentukan semua armada milik akun tertentu, gunakan wildcard (*):

```
"Resource": "arn:aws:gamelift:us-west-2:123456789012:fleet/*"
```

Untuk melihat contoh Amazon GameLift Servers Kebijakan berbasis identitas, lihat. [Contoh kebijakan berbasis identitas untuk Amazon GameLift Servers](#)

Kunci kondisi kebijakan untuk Amazon GameLift Servers

Mendukung kunci kondisi kebijakan khusus layanan: Yes

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Yaitu, di mana utama dapat melakukan tindakan pada sumber daya, dan dalam kondisi apa.

Elemen `Condition` (atau blok `Condition`) akan memungkinkan Anda menentukan kondisi yang menjadi dasar suatu pernyataan berlaku. Elemen `Condition` bersifat opsional. Anda dapat membuat ekspresi bersyarat yang menggunakan [operator kondisi](#), misalnya sama dengan atau kurang dari, untuk mencocokkan kondisi dalam kebijakan dengan nilai-nilai yang diminta.

Jika Anda menentukan beberapa elemen `Condition` dalam sebuah pernyataan, atau beberapa kunci dalam elemen `Condition` tunggal, maka AWS akan mengevaluasinya menggunakan operasi AND logis. Jika Anda menentukan beberapa nilai untuk satu kunci kondisi, AWS mengevaluasi kondisi menggunakan OR operasi logis. Semua kondisi harus dipenuhi sebelum izin pernyataan diberikan.

Anda juga dapat menggunakan variabel placeholder saat menentukan kondisi. Sebagai contoh, Anda dapat memberikan izin kepada pengguna IAM untuk mengakses sumber daya hanya jika izin tersebut mempunyai tanda yang sesuai dengan nama pengguna IAM mereka. Untuk informasi selengkapnya, lihat [Elemen kebijakan IAM: variabel dan tanda](#) dalam Panduan Pengguna IAM.

AWS mendukung kunci kondisi global dan kunci kondisi khusus layanan. Untuk melihat semua kunci kondisi AWS global, lihat [kunci konteks kondisi AWS global](#) di Panduan Pengguna IAM.

Untuk daftar Amazon GameLift Servers tombol kondisi, lihat tombol [Kondisi untuk Amazon GameLift Servers](#) dalam Referensi Otorisasi Layanan. Untuk mempelajari tindakan dan sumber daya yang dapat Anda gunakan kunci kondisi, lihat [Tindakan yang ditentukan oleh Amazon GameLift Servers](#).

Untuk melihat contoh Amazon GameLift Servers Kebijakan berbasis identitas, lihat. [Contoh kebijakan berbasis identitas untuk Amazon GameLift Servers](#)

ACLs di Amazon GameLift Servers

Mendukung ACLs: Tidak

Access control lists (ACLs) mengontrol prinsipal mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACLs mirip dengan kebijakan berbasis sumber daya, meskipun mereka tidak menggunakan format dokumen kebijakan JSON.

ABAC dengan Amazon GameLift Servers

Mendukung ABAC (tanda dalam kebijakan): Ya

Kontrol akses berbasis atribut (ABAC) adalah strategi otorisasi yang menentukan izin berdasarkan atribut. Dalam AWS, atribut ini disebut tag. Anda dapat melampirkan tag ke entitas IAM (pengguna atau peran) dan ke banyak AWS sumber daya. Penandaan ke entitas dan sumber daya adalah langkah pertama dari ABAC. Kemudian rancanglah kebijakan ABAC untuk mengizinkan operasi ketika tanda milik prinsipal cocok dengan tanda yang ada di sumber daya yang ingin diakses.

ABAC sangat berguna di lingkungan yang berkembang dengan cepat dan berguna di situasi saat manajemen kebijakan menjadi rumit.

Untuk mengendalikan akses berdasarkan tanda, berikan informasi tentang tanda di [elemen kondisi](#) dari kebijakan menggunakan kunci kondisi `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`, atau `aws:TagKeys`.

Jika sebuah layanan mendukung ketiga kunci kondisi untuk setiap jenis sumber daya, nilainya adalah Ya untuk layanan tersebut. Jika suatu layanan mendukung ketiga kunci kondisi untuk hanya beberapa jenis sumber daya, nilainya adalah Parsial.

Untuk informasi selengkapnya tentang ABAC, lihat [Tentukan izin dengan otorisasi ABAC](#) dalam Panduan Pengguna IAM. Untuk melihat tutorial yang menguraikan langkah-langkah pengaturan ABAC, lihat [Menggunakan kontrol akses berbasis atribut \(ABAC\)](#) dalam Panduan Pengguna IAM.

Untuk contoh kebijakan berbasis identitas yang membatasi akses ke sumber daya berdasarkan tag pada sumber daya tersebut, lihat. [Tayang Amazon GameLift Servers armada berdasarkan tag](#)

Menggunakan kredensi sementara dengan Amazon GameLift Servers

Mendukung kredensial sementara: Ya

Beberapa Layanan AWS tidak berfungsi saat Anda masuk menggunakan kredensi sementara. Untuk informasi tambahan, termasuk yang Layanan AWS bekerja dengan kredensi sementara, lihat [Layanan AWS yang bekerja dengan IAM di Panduan Pengguna IAM](#).

Anda menggunakan kredensi sementara jika Anda masuk AWS Management Console menggunakan metode apa pun kecuali nama pengguna dan kata sandi. Misalnya, ketika Anda mengakses AWS menggunakan tautan masuk tunggal (SSO) perusahaan Anda, proses tersebut secara otomatis membuat kredensial sementara. Anda juga akan secara otomatis membuat kredensial sementara ketika Anda masuk ke konsol sebagai seorang pengguna lalu beralih peran. Untuk informasi selengkapnya tentang peralihan peran, lihat [Beralih dari pengguna ke peran IAM \(konsol\)](#) dalam Panduan Pengguna IAM.

Anda dapat membuat kredensial sementara secara manual menggunakan API AWS CLI atau AWS . Anda kemudian dapat menggunakan kredensial sementara tersebut untuk mengakses AWS . AWS merekomendasikan agar Anda secara dinamis menghasilkan kredensial sementara alih-alih menggunakan kunci akses jangka panjang. Untuk informasi selengkapnya, lihat [Kredensial keamanan sementara di IAM](#).

Izin utama lintas layanan untuk Amazon GameLift Servers

Mendukung sesi akses maju (FAS): Ya

Saat Anda menggunakan pengguna atau peran IAM untuk melakukan tindakan AWS, Anda dianggap sebagai prinsipal. Ketika Anda menggunakan beberapa layanan, Anda mungkin melakukan sebuah tindakan yang kemudian menginisiasi tindakan lain di layanan yang berbeda. FAS menggunakan izin dari pemanggilan utama Layanan AWS, dikombinasikan dengan permintaan Layanan AWS untuk membuat permintaan ke layanan hilir. Permintaan FAS hanya dibuat ketika layanan menerima permintaan yang memerlukan interaksi dengan orang lain Layanan AWS atau sumber daya untuk menyelesaikannya. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan ketika mengajukan permintaan FAS, lihat [Sesi akses maju](#).

Peran layanan untuk Amazon GameLift Servers

Mendukung peran layanan: Ya

Peran layanan adalah [peran IAM](#) yang diambil oleh sebuah layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, mengubah, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat [Buat sebuah peran untuk mendelegasikan izin ke Layanan AWS](#) dalam Panduan pengguna IAM.

Warning

Mengubah izin untuk peran layanan mungkin rusak Amazon GameLift Servers fungsionalitas. Edit peran layanan hanya ketika Amazon GameLift Servers memberikan panduan untuk melakukannya.

Izinkan Anda Amazon GameLift Servers-host server game untuk mengakses AWS sumber daya lain, seperti AWS Lambda fungsi atau database Amazon DynamoDB. Karena server game di-host di armada itu Amazon GameLift Servers mengelola, Anda membutuhkan peran layanan yang memberi Amazon GameLift Servers akses terbatas ke AWS sumber daya Anda yang lain. Untuk informasi selengkapnya, lihat [Berkomunikasi dengan AWS sumber daya lain dari armada Anda](#).

Peran terkait layanan untuk Amazon GameLift Servers

Mendukung peran terkait layanan: Tidak

Peran terkait layanan adalah jenis peran layanan yang ditautkan ke. Layanan AWS Layanan tersebut dapat menjalankan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan muncul di Anda Akun AWS dan dimiliki oleh layanan. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran tertaut-layanan.

Untuk informasi selengkapnya tentang cara membuat atau mengelola peran yang tertaut dengan layanan, lihat [Layanan AWS yang bekerja dengan IAM](#) di Panduan Pengguna IAM. Temukan layanan dalam tabel yang menyertakan Yes dalam kolom Peran terkait layanan. Pilih Ya untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut.

Contoh kebijakan berbasis identitas untuk Amazon GameLift Servers

Secara default, pengguna dan peran tidak memiliki izin untuk membuat atau memodifikasi Amazon GameLift Servers sumber daya. Mereka juga tidak dapat melakukan tugas dengan menggunakan AWS Management Console, AWS Command Line Interface (AWS CLI), atau AWS API. Untuk memberikan izin kepada pengguna untuk melakukan tindakan di sumber daya yang mereka perlukan, administrator IAM dapat membuat kebijakan IAM. Administrator kemudian dapat menambahkan kebijakan IAM ke peran, dan pengguna dapat mengambil peran.

Untuk mempelajari cara membuat kebijakan berbasis identitas IAM dengan menggunakan contoh dokumen kebijakan JSON ini, lihat [Membuat kebijakan IAM \(konsol\) di Panduan Pengguna IAM](#).

Untuk detail tentang tindakan dan jenis sumber daya yang ditentukan oleh Amazon GameLift Servers, termasuk format ARNs untuk setiap jenis sumber daya, lihat [Tindakan, sumber daya, dan kunci kondisi untuk Amazon GameLift Servers](#) dalam Referensi Otorisasi Layanan.

Topik

- [Praktik terbaik kebijakan](#)
- [Menggunakan Amazon GameLift Servers konsol](#)
- [Mengizinkan pengguna melihat izin mereka sendiri](#)
- [Izinkan akses pemain untuk sesi game](#)
- [Izinkan akses ke satu Amazon GameLift Servers antrean](#)
- [Tayang Amazon GameLift Servers armada berdasarkan tag](#)
- [Mengakses file build game di Amazon S3](#)

Praktik terbaik kebijakan

Kebijakan berbasis identitas menentukan apakah seseorang dapat membuat, mengakses, atau menghapus Amazon GameLift Servers sumber daya di akun Anda. Tindakan ini membuat Akun AWS Anda dikenai biaya. Ketika Anda membuat atau mengedit kebijakan berbasis identitas, ikuti panduan dan rekomendasi ini:

- Mulailah dengan kebijakan AWS terkelola dan beralih ke izin hak istimewa paling sedikit — Untuk mulai memberikan izin kepada pengguna dan beban kerja Anda, gunakan kebijakan AWS terkelola yang memberikan izin untuk banyak kasus penggunaan umum. Mereka tersedia di Anda Akun AWS. Kami menyarankan Anda mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola AWS pelanggan yang khusus untuk kasus penggunaan Anda. Untuk informasi selengkapnya, lihat [Kebijakan yang dikelola AWS](#) atau [Kebijakan yang dikelola AWS untuk fungsi tugas](#) dalam Panduan Pengguna IAM.
- Menerapkan izin dengan hak akses paling rendah – Ketika Anda menetapkan izin dengan kebijakan IAM, hanya berikan izin yang diperlukan untuk melakukan tugas. Anda melakukannya dengan mendefinisikan tindakan yang dapat diambil pada sumber daya tertentu dalam kondisi tertentu, yang juga dikenal sebagai izin dengan hak akses paling rendah. Untuk informasi selengkapnya tentang cara menggunakan IAM untuk mengajukan izin, lihat [Kebijakan dan izin dalam IAM](#) dalam Panduan Pengguna IAM.
- Gunakan kondisi dalam kebijakan IAM untuk membatasi akses lebih lanjut – Anda dapat menambahkan suatu kondisi ke kebijakan Anda untuk membatasi akses ke tindakan dan sumber daya. Sebagai contoh, Anda dapat menulis kondisi kebijakan untuk menentukan bahwa semua permintaan harus dikirim menggunakan SSL. Anda juga dapat menggunakan ketentuan untuk memberikan akses ke tindakan layanan jika digunakan melalui yang spesifik Layanan AWS, seperti AWS CloudFormation. Untuk informasi selengkapnya, lihat [Elemen kebijakan JSON IAM: Kondisi](#) dalam Panduan Pengguna IAM.
- Gunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda untuk memastikan izin yang aman dan fungsional – IAM Access Analyzer memvalidasi kebijakan baru dan yang sudah ada sehingga kebijakan tersebut mematuhi bahasa kebijakan IAM (JSON) dan praktik terbaik IAM. IAM Access Analyzer menyediakan lebih dari 100 pemeriksaan kebijakan dan rekomendasi yang dapat ditindaklanjuti untuk membantu Anda membuat kebijakan yang aman dan fungsional. Untuk informasi selengkapnya, lihat [Validasi kebijakan dengan IAM Access Analyzer](#) dalam Panduan Pengguna IAM.
- Memerlukan otentikasi multi-faktor (MFA) - Jika Anda memiliki skenario yang mengharuskan pengguna IAM atau pengguna root di Anda, Akun AWS aktifkan MFA untuk keamanan tambahan.

Untuk meminta MFA ketika operasi API dipanggil, tambahkan kondisi MFA pada kebijakan Anda. Untuk informasi selengkapnya, lihat [Amankan akses API dengan MFA](#) dalam Panduan Pengguna IAM.

Untuk informasi selengkapnya tentang praktik terbaik dalam IAM, lihat [Praktik terbaik keamanan di IAM](#) dalam Panduan Pengguna IAM.

Menggunakan Amazon GameLift Servers konsol

Untuk mengakses Amazon GameLift Servers konsol, Anda harus memiliki set izin minimum. Izin ini harus memungkinkan Anda untuk membuat daftar dan melihat detail tentang Amazon GameLift Servers sumber daya di Anda Akun AWS. Jika Anda membuat kebijakan berbasis identitas yang lebih ketat daripada izin minimum yang diperlukan, konsol tidak akan berfungsi sebagaimana mestinya untuk entitas (pengguna atau peran) dengan kebijakan tersebut.

Untuk memastikan bahwa entitas tersebut masih dapat menggunakan Amazon GameLift Servers konsol, tambahkan izin ke pengguna dan grup dengan sintaks dalam contoh berikut dan di. [Contoh izin administrasi](#) Untuk informasi selengkapnya, lihat [Tetapkan izin pengguna untuk Amazon GameLift Servers](#).

Pengguna yang bekerja dengan Amazon GameLift Servers melalui AWS CLI atau operasi AWS API tidak memerlukan izin konsol minimum. Sebagai gantinya, Anda dapat membatasi akses hanya ke operasi yang perlu dilakukan pengguna. Misalnya, pengguna pemain, yang bertindak atas nama klien game, memerlukan akses untuk meminta sesi permainan, menempatkan pemain ke dalam game, dan tugas lainnya.

Untuk informasi tentang izin yang diperlukan untuk menggunakan semua Amazon GameLift Servers fitur konsol, lihat sintaks izin untuk administrator di. [Contoh izin administrasi](#)

Mengizinkan pengguna melihat izin mereka sendiri

Contoh ini menunjukkan cara membuat kebijakan yang mengizinkan pengguna IAM melihat kebijakan inline dan terkelola yang dilampirkan ke identitas pengguna mereka. Kebijakan ini mencakup izin untuk menyelesaikan tindakan ini di konsol atau menggunakan API atau secara terprogram. AWS CLI AWS

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

```

        "Sid": "ViewOwnUserInfo",
        "Effect": "Allow",
        "Action": [
            "iam:GetUserPolicy",
            "iam:ListGroupsForUser",
            "iam:ListAttachedUserPolicies",
            "iam:ListUserPolicies",
            "iam:GetUser"
        ],
        "Resource": ["arn:aws:iam::*:user/${aws:username}"]
    },
    {
        "Sid": "NavigateInConsole",
        "Effect": "Allow",
        "Action": [
            "iam:GetGroupPolicy",
            "iam:GetPolicyVersion",
            "iam:GetPolicy",
            "iam:ListAttachedGroupPolicies",
            "iam:ListGroupPolicies",
            "iam:ListPolicyVersions",
            "iam:ListPolicies",
            "iam:ListUsers"
        ],
        "Resource": "*"
    }
]
}

```

Izinkan akses pemain untuk sesi game

Untuk menempatkan pemain ke dalam sesi permainan, klien game dan layanan backend memerlukan izin. Untuk contoh kebijakan untuk skenario ini, lihat [Contoh izin pengguna pemain](#).

Izinkan akses ke satu Amazon GameLift Servers antrian

Contoh berikut menyediakan pengguna dengan akses ke tertentu Amazon GameLift Servers antrian.

Kebijakan ini memberikan izin pengguna untuk menambah, memperbarui, dan menghapus tujuan antrian dengan tindakan berikut: `gamelift:UpdateGameSessionQueue`, `gamelift>DeleteGameSessionQueue` dan `gamelift:DescribeGameSessionQueues` Seperti yang ditunjukkan, kebijakan ini menggunakan Resource elemen untuk membatasi akses ke satu antrian: `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"
    }
  ]
}
```

Tayang Amazon GameLift Servers armada berdasarkan tag

Anda dapat menggunakan kondisi dalam kebijakan berbasis identitas Anda untuk mengontrol akses Amazon GameLift Servers sumber daya berdasarkan tag. Contoh ini menunjukkan cara membuat kebijakan yang memungkinkan melihat armada jika Owner tag cocok dengan nama pengguna. Kebijakan ini juga memberikan izin yang diperlukan untuk menyelesaikan operasi ini di konsol.

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

Mengakses file build game di Amazon S3

Setelah Anda mengintegrasikan server game Anda dengan Amazon GameLift Servers, unggah file build ke Amazon S3. Untuk Amazon GameLift Servers untuk mengakses file build, gunakan kebijakan berikut.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion"
      ],
      "Resource": "arn:aws:s3:::bucket-name/object-name"
    }
  ]
}
```

Untuk informasi lebih lanjut tentang mengunggah Amazon GameLift Servers file game, lihat [Menerapkan build server khusus untuk Amazon GameLift Servers mengadakan](#).

Pemecahan Masalah Amazon GameLift Servers identitas dan akses

Gunakan informasi berikut untuk membantu Anda mendiagnosis dan memperbaiki masalah umum yang mungkin Anda temui saat bekerja dengan Amazon GameLift Servers dan AWS Identity and Access Management (IAM).

Topik

- [Saya tidak berwenang untuk melakukan tindakan di Amazon GameLift Servers](#)
- [Saya tidak berwenang untuk melakukan iam: PassRole](#)
- [Saya ingin mengizinkan orang-orang di luar saya Akun AWS untuk mengakses Amazon GameLift Servers sumber daya](#)

Saya tidak berwenang untuk melakukan tindakan di Amazon GameLift Servers

Jika AWS Management Console memberitahu Anda bahwa Anda tidak berwenang untuk melakukan tindakan, hubungi administrator AWS akun Anda untuk bantuan. Administrator Anda adalah orang yang memberi Anda kredensial masuk.

Contoh kesalahan berikut terjadi ketika pengguna mateojackson IAM mencoba menggunakan konsol untuk melihat detail tentang antrian tetapi tidak memiliki `gamelift:DescribeGameSessionQueues` izin:

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
gamelift:DescribeGameSessionQueues on resource: examplequeue123
```

Dalam hal ini, Mateo meminta administratornya untuk memperbarui kebijakannya untuk memungkinkannya membaca akses untuk `examplequeue123` sumber daya menggunakan `gamelift:DescribeGameSessionQueues` tindakan tersebut.

Saya tidak berwenang untuk melakukan iam: PassRole

Jika Anda menerima kesalahan yang tidak diizinkan untuk melakukan `iam:PassRole` tindakan, kebijakan Anda harus diperbarui agar Anda dapat meneruskan peran Amazon GameLift Servers.

Beberapa Layanan AWS memungkinkan Anda untuk meneruskan peran yang ada ke layanan tersebut alih-alih membuat peran layanan baru atau peran terkait layanan. Untuk melakukannya, Anda harus memiliki izin untuk meneruskan peran ke layanan.

Contoh kesalahan berikut terjadi ketika pengguna IAM bernama marymajor mencoba menggunakan konsol untuk melakukan tindakan di Amazon GameLift Servers. Namun, tindakan tersebut mengharuskan layanan memiliki izin yang diberikan oleh peran layanan. Mary tidak memiliki izin untuk meneruskan peran tersebut pada layanan.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
iam:PassRole
```

Dalam kasus ini, kebijakan Mary harus diperbarui agar dia mendapatkan izin untuk melakukan tindakan `iam:PassRole` tersebut.

Jika Anda memerlukan bantuan, hubungi AWS administrator Anda. Administrator Anda adalah orang yang memberi Anda kredensial masuk.

Saya ingin mengizinkan orang-orang di luar saya Akun AWS untuk mengakses Amazon GameLift Servers sumber daya

Anda dapat membuat peran yang dapat digunakan pengguna di akun lain atau orang-orang di luar organisasi Anda untuk mengakses sumber daya Anda. Anda dapat menentukan siapa saja yang dipercaya untuk mengambil peran tersebut. Untuk layanan yang mendukung kebijakan berbasis sumber daya atau daftar kontrol akses (ACLs), Anda dapat menggunakan kebijakan tersebut untuk memberi orang akses ke sumber daya Anda.

Untuk mempelajari selengkapnya, periksa referensi berikut:

- Untuk mengetahui apakah Amazon GameLift Servers mendukung fitur-fitur ini, lihat [Bagaimana Amazon GameLift Servers bekerja dengan IAM](#).
- Untuk mempelajari cara menyediakan akses ke sumber daya Anda di seluruh sumber daya Akun AWS yang Anda miliki, lihat [Menyediakan akses ke pengguna IAM di pengguna lain Akun AWS yang Anda miliki](#) di Panduan Pengguna IAM.
- Untuk mempelajari cara menyediakan akses ke sumber daya Anda kepada pihak ketiga Akun AWS, lihat [Menyediakan akses yang Akun AWS dimiliki oleh pihak ketiga](#) dalam Panduan Pengguna IAM.
- Untuk mempelajari cara memberikan akses melalui federasi identitas, lihat [Menyediakan akses ke pengguna terautentikasi eksternal \(federasi identitas\)](#) dalam Panduan Pengguna IAM.
- Untuk mempelajari perbedaan antara menggunakan peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat [Akses sumber daya lintas akun di IAM di Panduan Pengguna IAM](#).

AWS kebijakan terkelola untuk Amazon GameLift Servers

Kebijakan AWS terkelola adalah kebijakan mandiri yang dibuat dan dikelola oleh AWS. AWS Kebijakan terkelola dirancang untuk memberikan izin bagi banyak kasus penggunaan umum sehingga Anda dapat mulai menetapkan izin kepada pengguna, grup, dan peran.

Perlu diingat bahwa kebijakan AWS terkelola mungkin tidak memberikan izin hak istimewa paling sedikit untuk kasus penggunaan spesifik Anda karena tersedia untuk digunakan semua pelanggan. AWS Kami menyarankan Anda untuk mengurangi izin lebih lanjut dengan menentukan [kebijakan yang dikelola pelanggan](#) yang khusus untuk kasus penggunaan Anda.

Anda tidak dapat mengubah izin yang ditentukan dalam kebijakan AWS terkelola. Jika AWS memperbarui izin yang ditentukan dalam kebijakan AWS terkelola, pemutakhiran akan memengaruhi semua identitas utama (pengguna, grup, dan peran) yang dilampirkan kebijakan tersebut. AWS kemungkinan besar akan memperbarui kebijakan AWS terkelola saat baru Layanan AWS diluncurkan atau operasi API baru tersedia untuk layanan yang ada.

Untuk informasi selengkapnya, lihat [Kebijakan terkelola AWS](#) dalam Panduan Pengguna IAM.

AWS kebijakan terkelola: GameLiftContainerFleetPolicy

Anda dapat melampirkan GameLiftContainerFleetPolicy ke peran IAM Anda.

Kebijakan ini memberikan izin untuk tindakan komputasi dalam Amazon GameLift Servers armada kontainer. Armada kontainer adalah seperangkat sumber daya hosting yang Amazon GameLift Servers mengelola untuk Anda. Amazon GameLift Servers membutuhkan izin untuk terhubung ke Amazon GameLift Servers layanan dan AWS layanan lainnya atas nama Anda.

Saat membuat armada kontainer dengan Amazon GameLift Servers, menyediakan peran layanan IAM dengan kebijakan GameLiftContainerFleetPolicy terkelola terlampir. Untuk petunjuk tentang cara membuat peran layanan, lihat [Menyiapkan peran layanan IAM untuk Amazon GameLift Servers](#).

Untuk informasi selengkapnya, lihat [GameLiftContainerFleetPolicy](#).

Detail izin

Kebijakan ini mencakup izin berikut.

- `cloudwatch`— Memungkinkan Amazon GameLift Servers untuk menulis log sesi game ke aliran log CloudWatch Acara Amazon di AWS akun Anda.
- `cloudwatch` — Memungkinkan Amazon GameLift Servers untuk membuat grup CloudWatch log untuk mengatur data sesi game di aliran log.
- `s3`— Memungkinkan Amazon GameLift Servers untuk menulis log sesi game ke bucket Amazon Simple Storage Service di AWS akun Anda.
- `s3`— Memungkinkan Amazon GameLift Servers untuk mengambil Wilayah AWS tempat bucket Amazon S3 tertentu berada, menggunakan tindakan API. `s3:GetBucketLocation`

- `gamelift` — Memungkinkan Amazon GameLift Servers untuk mengambil token otentikasi yang memungkinkan server game yang dihosting untuk berkomunikasi dengan Amazon GameLift Servers layanan melalui AWS akun Anda.

Amazon GameLift Servers pembaruan kebijakan AWS terkelola

Melihat detail tentang pembaruan kebijakan AWS terkelola Amazon GameLift Servers karena layanan ini mulai melacak perubahan ini. Untuk peringatan otomatis tentang perubahan pada halaman ini, berlangganan umpan RSS di [Amazon GameLift Servers Halaman catatan rilis](#).

Perubahan	Deskripsi	Tanggal
GameLiftContainerFleetPolicy — Ubah	Amazon GameLift Servers menambahkan izin baru untuk mengambil bucket Amazon Wilayah AWS S3.	Februari 5, 2024
GameLiftContainerFleetPolicy – Kebijakan baru	Amazon GameLift Servers menambahkan izin baru untuk mengaktifkan wadah server game berjalan Amazon GameLift Servers armada yang dikelola.	November 12, 2024
Amazon GameLift Servers mulai melacak perubahan	Amazon GameLift Servers mulai melacak perubahan untuk kebijakan AWS terkelola nya.	November 12, 2024

Pencatatan dan pemantauan dengan Amazon GameLift Servers

Pemantauan merupakan bagian penting dari menjaga keandalan, ketersediaan, dan kinerja Amazon GameLift Servers dan AWS solusi Anda. Anda harus mengumpulkan data pemantauan dari semua bagian AWS solusi Anda sehingga Anda dapat lebih mudah men-debug kegagalan multi-titik jika terjadi.

AWS dan Amazon GameLift Servers menyediakan beberapa alat untuk memantau sumber daya hosting game Anda dan menanggapi potensi insiden.

CloudWatch Alarm Amazon

Menggunakan CloudWatch alarm Amazon, Anda menonton satu metrik selama periode waktu yang Anda tentukan. Jika metrik melebihi ambang batas tertentu, notifikasi akan dikirim ke topik Amazon SNS atau kebijakan Auto AWS Scaling. CloudWatch alarm dipicu ketika statusnya berubah dan dipertahankan untuk sejumlah periode tertentu, bukan dengan berada dalam keadaan tertentu. Untuk informasi selengkapnya, lihat [Monitor Amazon GameLift Servers dengan Amazon CloudWatch](#).

AWS CloudTrail Log

CloudTrail menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau AWS layanan di Amazon GameLift Servers. Dengan menggunakan informasi yang dikumpulkan oleh CloudTrail, Anda dapat menentukan permintaan yang dibuat Amazon GameLift Servers, alamat IP dari mana permintaan dibuat, siapa yang membuat permintaan, kapan dibuat, dan detail tambahan. Untuk informasi selengkapnya, lihat [Pencatatan log Amazon GameLift Servers Panggilan API dengan AWS CloudTrail](#).

Validasi kepatuhan untuk Amazon GameLift Servers

Amazon GameLift Servers tidak dalam lingkup program AWS kepatuhan apa pun.

Untuk mempelajari apakah an Layanan AWS berada dalam lingkup program kepatuhan tertentu, lihat [Layanan AWS di Lingkup oleh Program Kepatuhan Layanan AWS](#) dan pilih program kepatuhan yang Anda minati. Untuk informasi umum, lihat [Program AWS Kepatuhan Program AWS](#) .

Anda dapat mengunduh laporan audit pihak ketiga menggunakan AWS Artifact. Untuk informasi selengkapnya, lihat [Mengunduh Laporan di AWS Artifact](#) .

Tanggung jawab kepatuhan Anda saat menggunakan Layanan AWS ditentukan oleh sensitivitas data Anda, tujuan kepatuhan perusahaan Anda, dan hukum dan peraturan yang berlaku. AWS menyediakan sumber daya berikut untuk membantu kepatuhan:

- [Kepatuhan dan Tata Kelola Keamanan](#) – Panduan implementasi solusi ini membahas pertimbangan arsitektur serta memberikan langkah-langkah untuk menerapkan fitur keamanan dan kepatuhan.
- [Referensi Layanan yang Memenuhi Syarat HIPAA](#) — Daftar layanan yang memenuhi syarat HIPAA. Tidak semua memenuhi Layanan AWS syarat HIPAA.

- [AWS Sumber Daya AWS](#) — Kumpulan buku kerja dan panduan ini mungkin berlaku untuk industri dan lokasi Anda.
- [AWS Panduan Kepatuhan Pelanggan](#) - Memahami model tanggung jawab bersama melalui lensa kepatuhan. Panduan ini merangkum praktik terbaik untuk mengamankan Layanan AWS dan memetakan panduan untuk kontrol keamanan di berbagai kerangka kerja (termasuk Institut Standar dan Teknologi Nasional (NIST), Dewan Standar Keamanan Industri Kartu Pembayaran (PCI), dan Organisasi Internasional untuk Standardisasi (ISO)).
- [Mengevaluasi Sumber Daya dengan Aturan](#) dalam Panduan AWS Config Pengembang — AWS Config Layanan menilai seberapa baik konfigurasi sumber daya Anda mematuhi praktik internal, pedoman industri, dan peraturan.
- [AWS Security Hub](#)— Ini Layanan AWS memberikan pandangan komprehensif tentang keadaan keamanan Anda di dalamnya AWS. Security Hub menggunakan kontrol keamanan untuk sumber daya AWS Anda serta untuk memeriksa kepatuhan Anda terhadap standar industri keamanan dan praktik terbaik. Untuk daftar layanan dan kontrol yang didukung, lihat [Referensi kontrol Security Hub](#).
- [Amazon GuardDuty](#) — Ini Layanan AWS mendeteksi potensi ancaman terhadap beban kerja Akun AWS, kontainer, dan data Anda dengan memantau lingkungan Anda untuk aktivitas mencurigakan dan berbahaya. GuardDuty dapat membantu Anda mengatasi berbagai persyaratan kepatuhan, seperti PCI DSS, dengan memenuhi persyaratan deteksi intrusi yang diamanatkan oleh kerangka kerja kepatuhan tertentu.
- [AWS Audit Manager](#) Ini Layanan AWS membantu Anda terus mengaudit AWS penggunaan Anda untuk menyederhanakan cara Anda mengelola risiko dan kepatuhan terhadap peraturan dan standar industri.

Ketahanan di Amazon GameLift Servers

Jika Anda menggunakan Amazon GameLift Servers FleetIQ sebagai fitur mandiri dengan Amazon EC2, lihat [Keamanan di Amazon EC2](#) di Panduan EC2 Pengguna Amazon.

Infrastruktur AWS global dibangun di sekitar AWS Wilayah dan Zona Ketersediaan. AWS Wilayah menyediakan beberapa Availability Zone yang terpisah secara fisik dan terisolasi, yang terhubung dengan latensi rendah, throughput tinggi, dan jaringan yang sangat redundan. Dengan Zona Ketersediaan, Anda dapat merancang serta mengoperasikan aplikasi dan basis data yang secara otomatis melakukan fail over di antara zona tanpa gangguan. Zona Ketersediaan memiliki ketersediaan dan toleransi kesalahan yang lebih baik, dan dapat diskalakan dibandingkan infrastruktur pusat data tunggal atau multi tradisional.

Untuk informasi selengkapnya tentang AWS Wilayah dan Availability Zone, lihat [infrastruktur AWS global](#).

Selain infrastruktur AWS global, Amazon GameLift Servers menawarkan fitur-fitur berikut untuk membantu mendukung kebutuhan ketahanan data Anda:

- **Antrian multi-wilayah** — Amazon GameLift Servers antrian sesi permainan digunakan untuk menempatkan sesi permainan baru dengan sumber daya hosting yang tersedia. Antrian yang mencakup beberapa Wilayah mampu mengalihkan penempatan sesi game jika terjadi pemadaman wilayah. Untuk informasi lebih lanjut dan praktik terbaik dalam membuat antrian sesi game, lihat [Kustomisasi antrian sesi permainan](#).
- **Penskalaan kapasitas otomatis** - Menjaga kesehatan dan ketersediaan sumber daya hosting Anda dengan menggunakan Amazon GameLift Servers alat penskalaan. Alat-alat ini menyediakan berbagai opsi yang memungkinkan Anda menyesuaikan kapasitas armada agar sesuai dengan kebutuhan game dan pemain Anda. Untuk informasi selengkapnya tentang penskalaan, lihat [Menskalakan kapasitas hosting game dengan Amazon GameLift Servers](#).
- **Distribusi di seluruh instance** - Amazon GameLift Servers mendistribusikan lalu lintas masuk di beberapa instance, tergantung pada ukuran armada. Sebagai praktik terbaik, game dalam produksi harus memiliki beberapa instans untuk menjaga ketersediaan jika instans menjadi tidak sehat atau tidak responsif.
- **Penyimpanan Amazon S3** - Build server game dan skrip yang diunggah Amazon GameLift Servers disimpan di Amazon S3 menggunakan kelas penyimpanan Standar, yang menggunakan beberapa replikasi pusat data untuk meningkatkan ketahanan. Log sesi game juga disimpan di Amazon S3 menggunakan kelas penyimpanan Standar.

Keamanan infrastruktur di Amazon GameLift Servers

Jika Anda menggunakan Amazon GameLift Servers FleetIQ sebagai fitur mandiri dengan Amazon EC2, lihat [Keamanan di Amazon EC2](#) di Panduan EC2 Pengguna Amazon.

Sebagai layanan yang dikelola, Amazon GameLift Servers dilindungi oleh prosedur keamanan jaringan AWS global yang dijelaskan dalam [Amazon Web Services: Ikhtisar proses keamanan](#) whitepaper.

Anda menggunakan panggilan API yang AWS dipublikasikan untuk mengakses Amazon GameLift Servers melalui jaringan. Klien harus mendukung Keamanan Lapisan Pengangkutan (TLS) 1.2 atau versi yang lebih baru. Kami merekomendasikan TLS 1.3 atau versi yang lebih baru. Klien juga harus

mendukung cipher suite dengan perfect forward secrecy (PFS) seperti Ephemeral Diffie-Hellman (DHE) atau Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Sebagian besar sistem modern seperti Java 7 dan sistem yang lebih baru mendukung mode ini.

Selain itu, permintaan harus ditandatangani menggunakan ID kunci akses dan kunci akses rahasia yang terkait dengan prinsipal IAM. Atau Anda dapat menggunakan [AWS Security Token Service](#) (AWS STS) untuk menghasilkan kredensial keamanan sementara untuk menandatangani permintaan.

Bagian Amazon GameLift Servers Layanan menempatkan semua armada ke awan pribadi virtual Amazon (VPCs) sehingga setiap armada ada di area yang terisolasi secara logis di AWS Cloud. Anda dapat menggunakan Amazon GameLift Servers kebijakan untuk mengontrol akses dari titik akhir VPC tertentu atau spesifik. VPCs Secara efektif, ini mengisolasi akses jaringan ke suatu Amazon GameLift Servers sumber daya hanya dari VPC tertentu dalam jaringan. AWS Bila Anda membuat armada, Anda menentukan kisaran nomor port dan alamat IP. Rentang ini membatasi bagaimana lalu lintas masuk dapat mengakses server game host pada VPC armada. Gunakan praktik terbaik keamanan standar saat memilih pengaturan akses armada.

Analisis konfigurasi dan kerentanan di Amazon GameLift Servers

Jika Anda menggunakan Amazon GameLift Servers FleetIQ sebagai fitur mandiri dengan Amazon EC2, lihat [Keamanan di Amazon EC2](#) di Panduan EC2 Pengguna Amazon.

Konfigurasi dan kontrol IT merupakan tanggung jawab bersama antara AWS dan Anda, pelanggan kami. Untuk informasi lebih lanjut, lihat [model tanggung jawab AWS bersama](#). AWS menangani tugas-tugas keamanan dasar seperti sistem operasi tamu (OS) dan patching database, konfigurasi firewall, dan pemulihan bencana. Prosedur ini telah ditinjau dan disertifikasi oleh pihak ketiga yang sesuai. Untuk detail selengkapnya, lihat sumber daya berikut: [Amazon Web Services: Ikhtisar proses keamanan](#) (whitepaper).

Praktik terbaik keamanan berikut juga membahas konfigurasi dan analisis kerentanan di Amazon GameLift Servers:

- Pelanggan bertanggung jawab atas pengelolaan perangkat lunak yang digunakan untuk Amazon GameLift Servers contoh untuk hosting game. Secara khusus:
 - Perangkat lunak aplikasi server game yang disediakan pelanggan harus dipelihara, termasuk pembaruan dan patch keamanan. Untuk memperbarui perangkat lunak server game, unggah build baru ke Amazon GameLift Servers, buat armada baru untuk itu, dan arahkan lalu lintas ke armada baru.

- Basis Amazon Machine Image (AMI), yang mencakup sistem operasi, diperbarui hanya saat armada baru dibuat. Untuk melakukan patch, memperbarui, dan mengamankan sistem operasi dan aplikasi lain yang merupakan bagian dari AMI, lakukan daur ulang armada secara teratur, terlepas dari pembaruan server game.
- Pelanggan harus mempertimbangkan untuk memperbarui game mereka secara teratur dengan versi SDK terbaru, termasuk AWS SDK, Amazon GameLift Servers Server SDK, dan Amazon GameLift Servers SDK Klien untuk Server Realtime.

Praktik terbaik keamanan untuk Amazon GameLift Servers

Jika Anda menggunakan Amazon GameLift Servers FleetIQ fitur mandiri dengan Amazon EC2, lihat [Keamanan di Amazon EC2](#) di Panduan EC2 Pengguna Amazon.

Amazon GameLift Servers menyediakan sejumlah fitur keamanan untuk dipertimbangkan ketika Anda mengembangkan dan menerapkan kebijakan keamanan Anda sendiri. Praktik terbaik berikut adalah pedoman umum dan tidak mewakili solusi keamanan yang lengkap. Karena praktik terbaik ini mungkin tidak sesuai atau cukup untuk lingkungan Anda, anggap sebagai pertimbangan yang membantu dan bukan sebagai resep.

Jangan membuka port ke Internet

Kami sangat menyarankan agar tidak membuka port ke Internet karena hal itu menimbulkan risiko keamanan. Misalnya, jika Anda menggunakan [UpdateFleetPortSettings](#) untuk membuka port desktop jarak jauh seperti ini:

```
{
  "FleetId": "<fleet identifier>",
  "InboundPermissionAuthorizations": [
    {
      "FromPort": 3389,
      "IpRange": "0.0.0.0/0",
      "Protocol": "RDP",
      "ToPort": 3389
    }
  ]
}
```

maka Anda mengizinkan siapa pun di Internet untuk mengakses instance.

Sebagai gantinya, buka port dengan alamat IP atau rentang alamat tertentu. Misalnya, seperti ini:

```
{
  "FleetId": "<fleet identifier>",
  "InboundPermissionAuthorizations": [
    {
      "FromPort": 3389,
      "IpRange": "54.186.139.221/32",
      "Protocol": "TCP",
      "ToPort": 3389
    }
  ]
}
```

Pelajari selengkapnya

Untuk informasi selengkapnya tentang cara memanfaatkan penggunaan yang Amazon GameLift Servers lebih aman, lihat [pilar AWS Well-Architected Tool Keamanan](#).

Amazon GameLift Serverspanduan referensi

Bagian ini berisi dokumentasi referensi untuk digunakanAmazon GameLift Servers.

Topik

- [API Layanan untuk Amazon GameLift Servers](#)
- [Server SDK 5.x untuk Amazon GameLift Servers](#)
- [Server SDK untuk Amazon GameLift Servers versi 4 dan sebelumnya](#)
- [Amazon GameLift ServersAmazon GameLift ServersRealtimereferensi](#)
- [Acara penempatan sesi permainan](#)
- [Amazon GameLift ServersVersi AMI](#)
- [Titik akhir dan kuota Amazon GameLift Servers](#)
- [Amazon GameLift ServersSuar ping UDP](#)

API Layanan untuk Amazon GameLift Servers

Gunakan daftar berbasis tugas ini untuk menemukan operasi API saat membangun solusi hosting Amazon GameLift Servers game Anda dan fitur lainnya. AWS SDK menyertakan operasi ini di `aws.gamelift` namespace. [Unduh AWS SDK](#) atau [lihat dokumentasi referensi Amazon GameLift Servers API](#). Anda juga dapat menggunakan API dengan antarmuka baris AWS perintah (AWS CLI), seperti yang didokumentasikan dalam [referensi AWS CLI perintah](#).

API mencakup dua set operasi untuk hosting game terkelola:

- [Kelola sumber daya Amazon GameLift Servers hosting](#)
- [Memulai sesi game dan bergabung dengan pemain](#)

API Amazon GameLift Servers Layanan juga berisi operasi untuk digunakan dengan Amazon GameLift Servers alat dan solusi lain. Untuk daftar FleetIQ APIs, lihat [Operasi FleetIQ API](#). Untuk daftar FlexMatch APIs perjodohan, lihat Operasi [FlexMatchAPI](#).

Kelola sumber daya Amazon GameLift Servers hosting

Hubungi operasi ini untuk mengonfigurasi sumber daya hosting untuk server game Anda, skala kapasitas untuk memenuhi permintaan pemain, mendapatkan metrik kinerja dan penggunaan, dan

banyak lagi. Gunakan operasi API ini saat menghosting server game dengan Amazon GameLift Servers, termasuk Amazon GameLift Servers Realtime. Anda juga dapat bekerja di [Amazon GameLift Servers konsol](#) untuk sebagian besar tugas manajemen sumber daya, atau Anda dapat melakukan panggilan dengan alat AWS Command Line Interface (AWS CLI).

Siapkan server game untuk deployment

Unggah dan konfigurasi kode server game Anda dalam persiapan deployment dan peluncuran sumber daya hosting.

Mengelola build server game kustom

- [upload-build](#) — Unggah file build dari jalur lokal dan buat sumber daya build baru Amazon GameLift Servers. Operasi ini, tersedia sebagai AWS CLI perintah, adalah cara paling umum untuk mengunggah build server game.
- [CreateBuild](#) — Buat build baru menggunakan file yang disimpan di bucket Amazon S3.
- [ListBuilds](#) — Dapatkan daftar semua build yang diunggah ke suatu Amazon GameLift Servers wilayah.
- [DescribeBuild](#) — Ambil informasi yang terkait dengan build.
- [UpdateBuild](#) — Ubah metadata build, termasuk nama dan versi build.
- [DeleteBuild](#) — Hapus build dari Amazon GameLift Servers.

Kelola skrip Amazon GameLift Servers Realtime konfigurasi

- [CreateScript](#) — Unggah JavaScript file dan buat sumber daya Amazon GameLift Servers skrip baru.
- [ListScripts](#) — Dapatkan daftar semua Realtime skrip yang diunggah ke suatu Amazon GameLift Servers wilayah.
- [DescribeScript](#) — Mengambil informasi yang terkait dengan Realtime skrip.
- [UpdateScript](#) — Ubah metadata skrip dan unggah konten skrip yang direvisi.
- [DeleteScript](#) — Hapus Realtime skrip dari Amazon GameLift Servers.

Mengatur sumber daya komputasi untuk hosting

Konfigurasi sumber daya hosting dan terapkan dengan pembuatan server game atau skrip Realtime konfigurasi Anda.

Buat dan kelola armada

- [CreateFleet](#)— Konfigurasi dan gunakan Amazon GameLift Servers armada sumber daya komputasi baru untuk menjalankan server game Anda. Setelah di-deploy, server game secara otomatis diluncurkan sesuai konfigurasi dan siap untuk menjadi host sesi game.
- [ListFleets](#)— Dapatkan daftar semua armada di suatu Amazon GameLift Servers wilayah.
- [DeleteFleet](#)— Hapus armada yang tidak lagi menjalankan server game atau hosting pemain.
- Melihat/memperbarui lokasi armada.
 - [CreateFleetLocations](#)— Tambahkan lokasi terpencil ke armada yang ada yang mendukung beberapa lokasi
 - [DescribeFleetLocationAttributes](#)— Dapatkan daftar semua lokasi terpencil untuk armada dan lihat status saat ini dari setiap lokasi.
 - [DeleteFleetLocations](#)— Hapus lokasi terpencil dari armada yang mendukung beberapa lokasi.
- Melihat/memperbarui konfigurasi armada.
 - [DescribeFleetAttributes/UpdateFleetAttributes](#)— Lihat atau ubah metadata dan pengaturan armada untuk perlindungan sesi game dan batas pembuatan sumber daya.
 - [DescribeFleetPortSettings/UpdateFleetPortSettings](#)— Lihat atau ubah izin masuk (alamat IP dan rentang pengaturan port) yang diizinkan untuk armada.
 - [DescribeRuntimeConfiguration/UpdateRuntimeConfiguration](#)— Lihat atau ubah proses server apa (dan berapa banyak) untuk dijalankan pada setiap instance dalam armada.

Kelola kapasitas armada

- [Jelaskan EC2 InstanceLimits](#) - Ambil jumlah maksimum instans yang diizinkan untuk AWS akun saat ini dan tingkat penggunaan saat ini.
- [DescribeFleetCapacity](#)— Ambil pengaturan kapasitas saat ini untuk Wilayah asal armada.
- [DescribeFleetLocationCapacity](#)— Ambil pengaturan kapasitas saat ini untuk setiap lokasi armada multi-lokasi.
- [UpdateFleetCapacity](#)— Secara manual menyesuaikan pengaturan kapasitas untuk armada.
- Mengatur:
 - [PutScalingPolicy](#)— Aktifkan auto-scaling berbasis target atau buat kebijakan auto-scaling khusus, atau perbarui kebijakan yang ada.
 - [DescribeScalingPolicies](#)— Ambil kebijakan auto-scaling yang ada.
 - [DeleteScalingPolicy](#)— Hapus kebijakan auto-scaling dan hentikan agar tidak memengaruhi kapasitas armada.

- [StartFleetActions](#)— Mulai ulang kebijakan auto-scaling armada.
- [StopFleetActions](#)— Tangguhkan kebijakan auto-scaling armada.

Pantau aktivitas armada.

- [DescribeFleetUtilization](#)— Ambil statistik tentang jumlah proses server, sesi permainan, dan pemain yang saat ini aktif di armada.
- [DescribeFleetLocationUtilization](#)— Ambil statistik pemanfaatan untuk setiap lokasi dalam armada multi-lokasi.
- [DescribeFleetEvents](#)— Lihat peristiwa yang dicatat untuk armada selama rentang waktu tertentu.
- [DescribeGameSessions](#)— Ambil metadata sesi permainan, termasuk waktu berjalan game dan jumlah pemain saat ini.

Mengatur antrian untuk penempatan sesi permainan

Siapkan antrian multi-armada dan multi-wilayah untuk menempatkan sesi game dengan sumber daya hosting terbaik yang tersedia dalam hal biaya, latensi, dan ketahanan.

- [CreateGameSessionQueue](#)— Buat antrian untuk digunakan saat memproses permintaan untuk penempatan sesi game.
- [DescribeGameSessionQueues](#)— Ambil antrian sesi permainan yang ditentukan di suatu wilayah. Amazon GameLift Servers
- [UpdateGameSessionQueue](#)— Ubah konfigurasi antrian sesi permainan.
- [DeleteGameSessionQueue](#)— Hapus antrian sesi permainan dari wilayah tersebut.

Mengelola alias

Gunakan alias untuk mewakili armada Anda atau membuat tujuan alternatif terminal. Alias berguna saat melakukan transisi aktivitas game dari satu armada ke armada lainnya, seperti saat memperbarui build server game.

- [CreateAlias](#)— Tentukan alias baru dan secara opsional menetapkannya ke armada.
- [ListAliases](#)— Dapatkan semua alias armada yang ditentukan di suatu Amazon GameLift Servers wilayah.
- [DescribeAlias](#)— Mengambil informasi tentang alias yang ada.

- [UpdateAlias](#)— Ubah pengaturan untuk alias, seperti mengarahkannya dari satu armada ke armada lainnya.
- [DeleteAlias](#)— Hapus alias dari wilayah tersebut.
- [ResolveAlias](#)— Dapatkan ID armada yang ditunjuk alias tertentu.

Connect ke instans hosting terkelola

Melihat informasi tentang masing-masing instans dalam armada, atau meminta akses jarak jauh ke instans armada tertentu untuk pemecahan masalah.

- [DescribeInstances](#)— Dapatkan informasi tentang setiap instance dalam armada, termasuk ID instans, alamat IP, lokasi, dan status.
- [GetInstanceAccess](#)— Minta kredensial akses yang diperlukan untuk terhubung dari jarak jauh ke instance tertentu dalam armada.

Mengatur peering VPC

Buat dan kelola koneksi peering VPC antara sumber daya Amazon GameLift Servers hosting Anda dan sumber daya lainnya. AWS

- [CreateVpcPeeringAuthorization](#)— Otorisasi koneksi peering ke salah satu dari Anda. VPCs
- [DescribeVpcPeeringAuthorizations](#)— Ambil otorisasi koneksi peering yang valid.
- [DeleteVpcPeeringAuthorization](#)— Hapus otorisasi koneksi peering.
- [CreateVpcPeeringConnection](#)— Buat koneksi peering antara VPC untuk armada dan salah satu Amazon GameLift Servers armada Anda. VPCs
- [DescribeVpcPeeringConnections](#)— Ambil informasi tentang koneksi peering VPC yang aktif atau tertunda dengan armada. Amazon GameLift Servers
- [DeleteVpcPeeringConnection](#)— Hapus koneksi peering VPC dengan armada. Amazon GameLift Servers

Memulai sesi game dan bergabung dengan pemain

Hubungi operasi ini dari layanan backend untuk memulai sesi permainan baru, mendapatkan informasi tentang sesi permainan yang ada, dan bergabung dengan pemain ke sesi permainan. Operasi ini untuk digunakan dengan server game khusus yang di-host Amazon GameLift

Servers. Jika Anda menggunakan Amazon GameLift ServersRealtime, kelola sesi permainan menggunakan [Amazon GameLift ServersRealtime referensi API klien \(C #\)](#).

- Mulai sesi permainan baru untuk satu atau lebih pemain.
 - [StartGameSessionPlacement](#)— Mintalah Amazon GameLift Servers untuk menemukan sumber daya hosting terbaik yang tersedia dan memulai sesi permainan baru. Ini adalah metode yang disukai untuk membuat sesi game baru. Ini bergantung pada antrian sesi permainan untuk melacak ketersediaan hosting di beberapa wilayah, dan menggunakan FleetIQ algoritme untuk memprioritaskan penempatan berdasarkan latensi pemain, biaya hosting, lokasi, dll.
 - [DescribeGameSessionPlacement](#)— Dapatkan detail dan status pada permintaan penempatan.
 - [StopGameSessionPlacement](#)— Batalkan permintaan penempatan.
 - [CreateGameSession](#)— Mulai sesi permainan baru yang kosong di lokasi armada tertentu. Operasi ini memberi Anda kontrol yang lebih besar atas tempat memulai sesi permainan, daripada menggunakan FleetIQ untuk mengevaluasi opsi penempatan. Anda harus menambahkan pemain ke sesi game baru dalam langkah terpisah.
- Dapatkan pemain ke sesi permainan yang ada. Cari sesi game berjalan dengan slot pemain yang tersedia dan simpan untuk pemain baru.
 - [CreatePlayerSession](#)— Pesan slot terbuka bagi pemain untuk bergabung dengan sesi permainan.
 - [CreatePlayerSessions](#)— Cadangan slot terbuka untuk beberapa pemain untuk bergabung dengan sesi permainan.
- Bekerja dengan sesi permainan dan data sesi pemain. Kelola informasi tentang sesi game dan sesi pemain.
 - [SearchGameSessions](#)— Minta daftar sesi permainan aktif berdasarkan serangkaian kriteria pencarian.
 - [DescribeGameSessions](#)— Ambil metadata untuk sesi permainan tertentu, termasuk lamanya waktu aktif dan jumlah pemain saat ini.
 - [DescribeGameSessionDetails](#)— Ambil metadata, termasuk pengaturan perlindungan sesi permainan, untuk satu atau lebih sesi permainan.
 - [DescribePlayerSessions](#)— Dapatkan detail tentang aktivitas pemain, termasuk status, waktu bermain, dan data pemain.
 - [UpdateGameSession](#)— Ubah pengaturan sesi permainan, seperti jumlah pemain maksimum dan kebijakan bergabung.
 - [GetGameSessionLogUrl](#)— Dapatkan lokasi log yang disimpan untuk sesi permainan.

Server SDK 5.x untuk Amazon GameLift Servers

Bagian ini menyediakan dokumentasi referensi untuk server SDK 5.x untuk Amazon GameLift Servers. Server SDK menyediakan fungsionalitas inti yang digunakan server game Anda untuk berinteraksi dengan Amazon GameLift Servers layanan. Misalnya, server game Anda menerima permintaan dari layanan untuk memulai dan menghentikan sesi permainan dan menyediakan pembaruan status sesi permainan reguler ke layanan. Integrasikan server game Anda dengan SDK server sebelum Anda menerapkannya untuk hosting.

Gunakan referensi SDK server ini untuk mengintegrasikan server game multipemain khusus Anda untuk hosting. Amazon GameLift Servers Untuk panduan tentang proses integrasi, lihat [Tambahkan Amazon GameLift Servers ke server game Anda](#).

Versi utama terbaru dari SDK server untuk Amazon GameLift Servers adalah 5.x. Fitur hosting berikut memerlukan penggunaan versi 5.x:

- Amazon GameLift Servers Anywhere
- Amazon GameLift Servers plugin untuk Unreal Engine dan Unity

Topik

- [Pembaruan di server SDK 5 untuk Amazon GameLift Servers](#)
- [Migrasi ke server SDK 5.x untuk Amazon GameLift Servers](#)
- [Server C++ SDK 5.x untuk Amazon GameLift Servers -- Tindakan](#)
- [C# server SDK 5.x untuk Amazon GameLift Servers -- Tindakan](#)
- [Go server SDK untuk Amazon GameLift Servers -- Tindakan](#)
- [Server C++ \(Tidak nyata\) SDK 5.x untuk -- Tindakan Amazon GameLift Servers](#)

Pembaruan di server SDK 5 untuk Amazon GameLift Servers

Server game yang dihosting menggunakan SDK server Amazon GameLift Servers untuk berkomunikasi dengan Amazon GameLift Servers layanan untuk memulai dan mengelola sesi permainan bagi pemain. Versi terbaru, Amazon GameLift Servers server SDK 5, menawarkan sejumlah perbaikan dan dukungan untuk Amazon GameLift Servers fitur-fitur baru. Jika pembuatan server game Anda saat ini menggunakan Amazon GameLift Servers server SDK 4 atau versi lebih lama, ikuti panduan dalam topik ini untuk memperbarui game Anda.

Amazon GameLift Servers server SDK versi 5.0.0 dan di atasnya mencakup pembaruan ini:

- Bahasa yang diperluas — Perpustakaan tersedia dalam bahasa berikut: C ++, C #, Go. Anda dapat membangun pustaka C++ untuk digunakan dengan Unreal Engine.
- Dukungan plugin mesin game - Plugin Amazon GameLift Servers mandiri untuk Unreal Engine dan Unity memerlukan pustaka SDK 5 Amazon GameLift Servers server. Plugin ini menawarkan alur kerja terpandu untuk mengintegrasikan, menguji, dan menyebarkan game Anda untuk hosting. Amazon GameLift Servers Lihat [Amazon GameLift Servers plugin untuk Unity \(server SDK 5.x\)](#) dan [Amazon GameLift Servers plugin untuk Unreal Engine](#) dokumentasi.
- Amazon GameLift Servers Dukungan di mana saja - Dengan armada Anywhere Anda dapat mengatur sumber daya hosting Anda sendiri untuk menggunakan Amazon GameLift Servers fitur (termasuk perijinan). Tambahkan Amazon GameLift Servers Agen untuk mengotomatiskan manajemen siklus hidup sesi game. Gunakan armada Anywhere untuk hosting produksi dengan perangkat keras lokal, atau siapkan lingkungan pengujian untuk pengembangan game berulang yang cepat. Lihat [Hosting di mana saja](#) dan [Amazon GameLift Servers Agen](#).
- Alat pengujian yang diperbarui — Fitur Amazon GameLift Servers Anywhere memungkinkan Anda mengatur lingkungan pengujian lokal atau berbasis cloud untuk game Anda. Siapkan pengujian dengan atau tanpa Amazon GameLift Servers Agen. Alat-alat ini menggantikan Amazon GameLift Servers Lokal. Lihat [Siapkan pengujian lokal dengan Amazon GameLift Servers Anywhere](#).
- Solusi .NET terkonsolidasi untuk C # - Server C # SDK 5.1+ mendukung .NET Framework 4.6.2 (ditingkatkan dari 4.6.1) dan .NET 6.0 dalam satu solusi. .NET Standard 2.1 tersedia dengan pustaka Unity-built.
- Compute Sumber daya baru - Sumber daya baru ini menggabungkan berbagai jenis sumber daya hosting. Ini termasuk sumber daya hosting berbasis cloud (armada terkelola EC2 atau kontainer) dan sumber daya hosting yang dikendalikan pelanggan (armada Anywhere). Ini termasuk pembaruan berikut:
 - Panggilan API baru untuk Compute sumber daya meliputi: [ListCompute\(\)](#), [DescribeCompute\(\)](#), dan [GetComputeAccess\(\)](#). Tindakan ini mengembalikan informasi sumber daya hosting untuk semua jenis Amazon GameLift Servers armada. Secara umum, untuk armada dengan server game yang menggunakan server SDK 5.x, gunakan tindakan khusus komputasi untuk mengganti tindakan khusus instance. Selain itu, tindakan ini untuk digunakan dalam armada Anywhere tanpa Amazon GameLift Servers Agen: [RegisterCompute\(\)](#), [DeregisterCompute\(\)](#), dan [GetComputeAuthToken\(\)](#).
 - Metrik baru ActiveCompute dengan CloudWatch dimensi FleetId, Location, dan ComputeType. Metrik ini menggantikan metrik ActiveInstances sebelumnya.

- Amazon EC2 Systems Manager (SSM) untuk akses jarak jauh — Untuk keamanan tambahan, gunakan SSM alih-alih SSH saat menghubungkan ke instans di armada terkelola. Amazon GameLift Servers Lihat [Terhubung dari jarak jauh ke Amazon GameLift Servers contoh armada](#).

Migrasi ke server SDK 5.x untuk Amazon GameLift Servers

Untuk memperbarui proyek game agar menggunakan server SDK versi 5.x, buat perubahan berikut:

1. Dapatkan SDK server terbaru untuk Amazon GameLift Servers paket untuk lingkungan pengembangan Anda [[Unduh situs](#)]. Ikuti petunjuk penginstalan dalam Readme file untuk paket dan versi yang Anda unduh. Lihat petunjuk ini untuk menggunakan server SDKs dengan proyek game Anda.
 - [Untuk lingkungan pengembangan menggunakan C ++, C #, atau Go](#)
 - [Untuk proyek Unreal Engine \(SDK server C++ hanya untuk pustaka Unreal\)](#)
 - [Untuk proyek Unity \(SDK server C# hanya untuk pustaka SDK for Unity\)](#)
 - [Untuk digunakan dengan Amazon GameLift Servers plugin untuk Unreal Engine](#)
 - [Untuk digunakan dengan Amazon GameLift Servers plugin untuk Unity](#)
2. Perbarui kode server Anda sebagai berikut:
 - Ubah fungsi callback kode server `onCreateGameSession()` menjadi `onStartGameSession()`.
 - Perbarui `InitSDK()` input yang sesuai:
 - Jika Anda berencana untuk menyebarkan build server game ke armada Amazon GameLift Servers terkelola atau EC2 armada Anywhere dengan Amazon GameLift Servers Agen:

Panggilan `InitSDK()` tanpa parameter ([C ++](#)) ([C #](#)) ([Tidak](#) nyata). Panggilan ini mengatur lingkungan komputasi dan WebSocket koneksi ke Amazon GameLift Servers layanan.
 - Jika Anda berencana untuk menyebarkan build server game ke armada Anywhere tanpa Amazon GameLift Servers Agen:

Panggilan `InitSDK()` dengan parameter server ([C ++](#)) ([C #](#)) ([Tidak](#) nyata). Proses server game menggunakan parameter ini untuk membuat koneksi dengan Amazon GameLift Servers layanan.
3. Jika build server game atau aplikasi host lainnya berkomunikasi dengan AWS sumber daya lain saat berjalan, Anda harus mengubah cara aplikasi mendapatkan akses ke sumber daya

tersebut. Ganti penggunaan `AssumeRoleCredentials` dengan aksi SDK server baru `GetFleetRoleCredentials()` (untuk server game) atau gunakan kredensial bersama (untuk aplikasi lain). Untuk informasi lebih lanjut tentang cara menerapkan perubahan ini, lihat [Berkomunikasi dengan AWS sumber daya lain dari armada Anda](#).

4. [Jika proyek Anda memanggil tindakan SDK server `GetInstanceCertificate\(\)` untuk mengambil sertifikat TLS, ubah kode Anda untuk menggunakan `new GetComputeCertificate\(\)` \(C++\) \(C#\) \(Unreal\) sebagai gantinya.](#)
5. Saat mengunggah build game Anda Amazon GameLift Servers (seperti dengan [upload-build](#) atau [CreateBuild\(\)](#)), setel `ServerSdkVersion` parameter ke versi 5.x yang Anda gunakan (parameter ini saat ini default ke 4.0.2). Parameter ini harus cocok dengan pustaka SDK server yang sebenarnya di build server game. Jika Anda menentukan versi yang salah untuk build server game yang diunggah, armada apa pun yang Anda buat dengan build tersebut akan gagal. Lihat [Menerapkan build server khusus untuk Amazon GameLift Servers mengadakan](#).

Contoh berikut menggambarkan cara menentukan versi SDK server:

```
aws gamelift upload-build \  
  --operating-system AMAZON_LINUX_2023 \  
  --server-sdk-version "5.0.0" \  
  --build-root "~/mygame" \  
  --name "My Game Nightly Build" \  
  --build-version "build 255" \  
  --region us-west-2
```

6. Jika Anda menggunakan skrip untuk terhubung dari jarak jauh ke armada terkelola, perbarui skrip untuk menggunakan proses baru, seperti yang dijelaskan dalam. [Terhubung dari jarak jauh ke Amazon GameLift Servers contoh armada](#)

Server C ++ SDK 5.x untuk Amazon GameLift Servers -- Tindakan

Gunakan referensi SDK 5.x server untuk mengintegrasikan game multipemain Anda untuk hosting. Amazon GameLift Servers Untuk panduan tentang proses integrasi, lihat [Tambahkan Amazon GameLift Servers ke server game Anda](#).

 Note

Topik ini menjelaskan Amazon GameLift Servers C++ API yang dapat Anda gunakan saat membangun dengan C++ Standard Library (`std`). Secara khusus, dokumentasi ini berlaku untuk kode yang Anda kompilasi dengan `-DDGAMELIFT_USE_STD=1` opsi.

Server C ++ SDK 5.x untuk Amazon GameLift Servers -- Tipe data

Gunakan referensi SDK 5.x server Amazon GameLift Servers C ++ untuk mengintegrasikan game multipemain Anda untuk hosting. Amazon GameLift Servers Untuk panduan tentang proses integrasi, lihat [Tambahkan Amazon GameLift Servers ke server game Anda](#).

 Note

Topik ini menjelaskan Amazon GameLift Servers C++ API yang dapat Anda gunakan saat membangun dengan C++ Standard Library (`std`). Secara khusus, dokumentasi ini berlaku untuk kode yang Anda kompilasi dengan `-DDGAMELIFT_USE_STD=1` opsi.

Server C ++ SDK 5.x untuk Amazon GameLift Servers -- Tindakan

Jenis Data

- [LogParameters](#)
- [ProcessParameters](#)
- [UpdateGameSession](#)
- [GameSession](#)
- [ServerParameters](#)
- [StartMatchBackfillRequest](#)
- [Pemain](#)
- [DescribePlayerSessionsRequest](#)
- [StopMatchBackfillRequest](#)
- [AttributeValue](#)
- [GetFleetRoleCredentialsRequest](#)

- [AwsLongOutcome](#)
- [AwsStringOutcome](#)
- [DescribePlayerSessionsOutcome](#)
- [DescribePlayerSessionsResult](#)
- [GenericOutcome](#)
- [GenericOutcomeCallable](#)
- [PlayerSession](#)
- [StartMatchBackfillOutcome](#)
- [StartMatchBackfillResult](#)
- [GetComputeCertificateOutcome](#)
- [GetComputeCertificateResult](#)
- [GetFleetRoleCredentialsOutcome](#)
- [GetFleetRoleCredentialsResult](#)
- [Init SDKOutcome](#)
- [GameLiftError](#)
- [Enum](#)

LogParameters

Objek yang mengidentifikasi file yang dihasilkan selama sesi permainan yang Amazon GameLift Servers ingin Anda unggah dan simpan setelah sesi permainan berakhir. Server game menyediakan LogParameters Amazon GameLift Servers sebagai bagian dari ProcessParameters objek dalam [ProcessReady\(\)](#) panggilan.

Sifat-sifat	Deskripsi
LogPaths	Daftar jalur direktori ke file log server game yang ingin Anda simpan Amazon GameLift Servers untuk akses future. Proses server menghasilkan file-file ini selama setiap sesi permainan. Anda menentukan jalur dan nama file di server game Anda dan menyimpannya di direktori build game root.

Jalur log harus absolut. Misalnya, jika build game Anda menyimpan log sesi game di jalur seperti `MyGame\sessionLogs\`, maka jalurnya akan berada `c:\game\MyGame\sessionLogs` di instance Windows.

Jenis: `std::vector<std::string>`

Wajib: Tidak

ProcessParameters

Tipe data ini berisi kumpulan parameter yang dikirim ke Amazon GameLift Servers dalam file [ProcessReady\(\)](#).

Sifat-sifat	Deskripsi
LogParameters	Objek dengan jalur direktori ke file yang dihasilkan selama sesi permainan. Amazon GameLift Servers menyalin dan menyimpan file untuk akses future. Jenis: <code>Aws::GameLift::Server:: LogParameters</code> Wajib: Tidak
OnHealthCheck	Fungsi callback yang Amazon GameLift Servers memanggil untuk meminta laporan status kesehatan dari proses server. Amazon GameLift Servers memanggil fungsi ini setiap 60 detik dan menunggu 60 detik untuk respons. Proses server kembali TRUE jika sehat, FALSE jika tidak sehat. Jika tidak ada respons yang dikembalikan, Amazon GameLift Servers mencatat proses server sebagai tidak sehat.

Jenis: `std::function<bool()>`
`onHealthCheck`

Wajib: Tidak

OnProcessTerminate

Fungsi callback yang Amazon GameLift Servers memanggil untuk memaksa proses server dimatikan. Setelah memanggil fungsi ini, Amazon GameLift Servers tunggu 5 menit hingga proses server dimatikan dan merespons dengan [ProcessEnding\(\)](#) panggilan sebelum mematikan proses server.

Jenis: `std::function<void()>`
`onProcessTerminate`

Wajib: Ya

OnStartGameSession

Fungsi callback yang Amazon GameLift Servers memanggil untuk mengaktifkan sesi permainan baru. Amazon GameLift Servers memanggil fungsi ini dalam menanggapi permintaan klien [CreateGameSession](#). Fungsi callback melewati [GameSession](#) objek.

Jenis: `const std::function<void (Aws::GameLift::Model::GameSession)>` `onStartGameSession`

Wajib: Ya

OnUpdateGameSession	Fungsi callback yang Amazon GameLift Servers memanggil untuk meneruskan objek sesi game yang diperbarui ke proses server. Amazon GameLift Servers memanggil fungsi ini ketika permintaan pengisian ulang kecocokan telah diproses untuk menyediakan data mak comblang yang diperbarui. Ini melewati GameSession objek, pembaruan status (updateReason), dan ID tiket isi ulang pertandingan. Jenis: <code>std::function<void(Aws::GameLift::Server::Model::UpdateGameSession)> onUpdateGameSession</code> Wajib: Tidak
Port	Nomor port yang didengarkan oleh proses server untuk koneksi pemain baru. Nilai harus masuk ke dalam kisaran port yang dikonfigurasi untuk setiap armada yang men-deploy build server game ini. Nomor port ini termasuk dalam sesi game dan objek sesi pemain, yang digunakan sesi game saat menghubungkan ke proses server. Jenis: Integer Wajib: Ya

UpdateGameSession

Jenis data ini diperbarui ke objek sesi permainan, yang mencakup alasan bahwa sesi permainan diperbarui dan ID tiket isi ulang terkait jika pengisian ulang digunakan untuk mengisi sesi pemain dalam sesi permainan.

Properti	Deskripsi
GameSession	Sebuah objek GameSession. GameSession Objek berisi properti yang menggambarkan sesi permainan. Jenis: <code>Aws::GameLift::Server::GameSession</code> Wajib: Ya
UpdateReason	Alasan bahwa sesi permainan sedang diperbarui. Jenis: <code>Aws::GameLift::Server::UpdateReason</code> Wajib: Ya
BackfillTicketId	ID tiket isi ulang yang mencoba memperbarui sesi permainan. Jenis: <code>std::string</code> Wajib: Tidak

GameSession

Tipe data ini memberikan detail sesi permainan.

Properti	Deskripsi
GameSessionId	Pengenal unik untuk sesi permainan. Sesi permainan ARN memiliki format berikut: <code>arn:aws:gamelift:<region>::gamesession/<fleet ID>/<custom ID string or idempotency token></code> Jenis: <code>std::string</code>

Properti	Deskripsi
	Wajib: Tidak
Nama	Label deskriptif dari sesi permainan. Jenis: <code>std::string</code> Wajib: Tidak
FleetId	Pengenal unik untuk armada tempat sesi permainan berjalan. Jenis: <code>std::string</code> Wajib: Tidak
MaximumPlayerSessionCount	Jumlah maksimum koneksi pemain ke sesi permainan. Jenis: <code>int</code> Wajib: Tidak
Port	Nomor port untuk sesi permainan. Untuk terhubung ke server Amazon GameLift Servers game, aplikasi memerlukan alamat IP dan nomor port. Jenis: <code>int</code> Wajib: Tidak
IpAddress	Alamat IP dari sesi game. Untuk terhubung ke server Amazon GameLift Servers game, aplikasi memerlukan alamat IP dan nomor port. Jenis: <code>std::string</code> Wajib: Tidak

Properti	Deskripsi
GameSessionData	Set properti sesi game khusus, diformat sebagai nilai string tunggal. Jenis: <code>std::string</code> Wajib: Tidak
MatchmakerData	Informasi tentang proses perjodohan yang digunakan untuk membuat sesi permainan, dalam sintaks JSON, diformat sebagai string. Selain konfigurasi perjodohan yang digunakan , ini berisi data tentang semua pemain yang ditugaskan untuk pertandingan, termasuk atribut pemain dan tugas tim. Jenis: <code>std::string</code> Wajib: Tidak
GameProperties	Satu set properti kustom untuk sesi permainan, diformat sebagai pasangan key:value. Properti ini diteruskan dengan permintaan untuk memulai sesi permainan baru. Jenis: <code>std :: vector < GameProperty ></code> Wajib: Tidak

Properti	Deskripsi
DnsName	Pengenal DNS yang ditetapkan ke instance yang menjalankan sesi permainan. Nilai memiliki format berikut: - Armada yang mendukung TLS:<unique identifier>.<region identifier>.amazongamelift.com . - Non-TLS-enabled armada:ec2-<unique identifier>.compute.amazonaws.com . Saat menghubungkan ke sesi permainan yang berjalan pada armada yang mendukung TLS, Anda harus menggunakan nama DNS, bukan alamat IP. Jenis: <code>std::string</code> Wajib: Tidak

ServerParameters

Informasi yang digunakan oleh proses server game untuk membuat koneksi dengan Amazon GameLift Servers layanan. Sertakan parameter ini saat menelepon [InitSDK\(\)](#) hanya jika build server game akan dikerahkan ke armada Anywhere atau armada kontainer tanpa Amazon GameLift Servers Agen. Untuk semua skenario penerapan lainnya, panggil [InitSDK\(\)](#) tanpa parameter.

Properti	Deskripsi
webSocketUrl	GameLiftServerSdkEndpoint Amazon GameLift ServersPengembalian ketika Anda RegisterCompute untuk sumber daya komputasi Amazon GameLift Servers Anywhere.

Properti	Deskripsi
	Jenis: <code>std::string</code> Wajib: Ya
ProsesSid	Pengenal unik yang terdaftar pada proses server yang menghosting game Anda. Jenis: <code>std::string</code> Wajib: Ya
hostId	HostIDIni adalah yang ComputeName digunakan saat Anda mendaftarkan komputasi Anda. Untuk informasi selengkapnya, lihat RegisterCompute . Jenis: <code>std::string</code> Wajib: Ya
FleeTID	Pengidentifikasi unik armada tempat komputasi terdaftar. Untuk informasi selengkapnya, lihat RegisterCompute . Jenis: <code>std::string</code> Wajib: Ya
AuthToken	Token otentikasi yang dihasilkan oleh Amazon GameLift Servers itu mengotentikasi server Anda. Amazon GameLift Servers Untuk informasi selengkapnya, lihat GetComputeAuthToken . Jenis: <code>std::string</code> Wajib: Ya

StartMatchBackfillRequest

Informasi yang digunakan untuk membuat permintaan pengisian ulang perjodohan. Server game mengkomunikasikan informasi ini Amazon GameLift Servers dalam [StartMatchBackfill\(\)](#) panggilan.

Properti	Deskripsi
GameSessionArn	Pengidentifikasi sesi permainan yang unik. Operasi API GetGameSessionId mengembalikan pengenal dalam format ARN. Jenis: <code>std::string</code> Wajib: Ya
MatchmakingConfigurationArn	Pengidentifikasi unik, dalam bentuk ARN, untuk digunakan mak comblang untuk permintaan ini. ARN mak comblang untuk sesi permainan asli ada di objek sesi permainan di properti data mak comblang. Pelajari selengkapnya tentang data matchmaker di Bekerja dengan data matchmaker. Jenis: <code>std::string</code> Wajib: Ya
Pemain	Satu set data yang mewakili semua pemain yang berada di sesi permainan. Matchmaker menggunakan informasi ini untuk mencari pemain baru yang cocok untuk pemain saat ini. Jenis: <code>std::vector<Player></code> Wajib: Ya
TicketId	Pengenal unik untuk tiket permintaan pencocokan atau pencocokan isi ulang. Jika Anda tidak memberikan nilai, Amazon GameLift Servers hasilkan satu. Gunakan pengident

Properti	Deskripsi
	ifikasi ini untuk melacak status tiket backfill match atau membatalkan permintaan jika diperlukan. Jenis: <code>std::string</code> Wajib: Tidak

Pemain

Tipe data ini mewakili pemain dalam perjodohan. Saat memulai permintaan perjodohan, pemain memiliki ID pemain, atribut, dan mungkin data latensi. Amazon GameLift Servers menambahkan informasi tim setelah pertandingan dibuat.

Properti	Deskripsi
LatencyInMS	Satu set nilai yang dinyatakan dalam milidetik yang menunjukkan jumlah latensi yang dialami pemain saat terhubung ke suatu lokasi. Jika properti ini digunakan, pemain hanya cocok untuk lokasi yang terdaftar. Jika mak comblang memiliki aturan yang mengevaluasi latensi pemain, pemain harus melaporkan latensi untuk dicocokkan. Jenis: <code>Dictionary<string,int></code> Wajib: Tidak
PlayerAttributes	Kumpulan pasangan kunci:nilai yang berisi informasi pemain untuk digunakan dalam perjodohan. Kunci atribut pemain harus cocok dengan yang PlayerAttributes digunakan dalam set aturan perjodohan.

Properti	Deskripsi
	Untuk informasi selengkapnya tentang atribut pemain, lihat AttributeValue. Jenis: <code>std::map<std::string, AttributeValue></code> Wajib: Tidak
PlayerId	Pengenal unik untuk pemain. Jenis: <code>std::string</code> Wajib: Tidak
Tim	Nama tim yang ditugaskan pemain dalam pertandingan. Anda menentukan nama tim dalam set aturan perjodohan. Jenis: <code>std::string</code> Wajib: Tidak

DescribePlayerSessionsRequest

Objek yang menentukan sesi pemain mana yang akan diambil. Proses server menyediakan informasi ini dengan [DescribePlayerSessions\(\)](#) panggilan keAmazon GameLift Servers.

Properti	Deskripsi
GameSessionId	Pengidentifikasi sesi permainan yang unik. Gunakan parameter ini untuk meminta semua sesi pemain untuk sesi game yang ditentukan. Format ID sesi game adalah <code>arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string></code>. GameSessionID Ini adalah string ID kustom atau

Properti	Deskripsi
	Jenis: <code>std::string</code> Wajib: Tidak
PlayerSessionId	Pengidentifikasi unik untuk sesi pemain. Gunakan parameter ini untuk meminta satu sesi pemain tertentu. Jenis: <code>std::string</code> Wajib: Tidak
PlayerId	Pengenal unik untuk pemain. Gunakan parameter ini untuk meminta semua sesi pemain untuk pemain tertentu. Lihat Hasilkan pemain IDs . Jenis: <code>std::string</code> Wajib: Tidak

Properti	Deskripsi
PlayerSessionStatusFilter	Status sesi pemain untuk memfilter hasil. Status sesi pemain yang mungkin meliputi: • RESERVED — Permintaan sesi pemain diterima, tetapi pemain belum terhubung ke proses server atau telah divalidasi. • AKTIF — Pemain divalidasi oleh proses server dan terhubung. • SELESAI — Koneksi pemain terputus. • TIMEDOUT — Permintaan sesi pemain diterima, tetapi pemain tidak terhubung atau tidak divalidasi dalam batas waktu habis (60 detik). Jenis: <code>std::string</code> Wajib: Tidak
NextToken	Token yang menunjukkan awal halaman hasil berikutnya. Untuk menentukan awal kumpulan hasil, jangan berikan nilai. Jika Anda memberikan ID sesi pemain, parameter ini diabaikan. Jenis: <code>std::string</code> Wajib: Tidak
Kuota	Jumlah hasil maksimum yang akan dikembalikan. Jika Anda memberikan ID sesi pemain, parameter ini diabaikan. Jenis: <code>int</code> Wajib: Tidak

StopMatchBackfillRequest

Informasi yang digunakan untuk membatalkan permintaan pengisian ulang perjodohan. Server game mengkomunikasikan informasi ini ke Amazon GameLift Servers layanan dalam [StopMatchBackfill\(\)](#) panggilan.

Properti	Deskripsi
GameSessionArn	Pengidentifikasi sesi permainan unik dari permintaan yang dibatalkan. Jenis: <code>char[]</code> Wajib: Tidak
MatchmakingConfigurationArn	Pengidentifikasi unik dari mak comblang permintaan ini dikirim ke. Jenis: <code>char[]</code> Wajib: Tidak
TicketId	Pengenal unik dari tiket permintaan isi ulang yang akan dibatalkan. Jenis: <code>char[]</code> Wajib: Tidak

AttributeValue

Gunakan nilai-nilai ini dalam [Pemain](#) atribut pasangan kunci-nilai. Objek ini memungkinkan Anda menentukan nilai atribut menggunakan salah satu tipe data yang valid: string, nomor, array string, atau peta data. Setiap `AttributeValue` objek harus menggunakan persis salah satu properti yang tersedia: `S`, `N`, `SL`, atau `SDM`.

Properti	Deskripsi
AttrType	Menentukan jenis nilai atribut. Jenis nilai atribut yang mungkin meliputi:

Properti	Deskripsi
	• TIDAK ADA • STRING • GANDA • STRING_LIST • STRING_DOUBLE_PETA Wajib: Tidak
D	Merupakan nilai atribut string. Jenis: <code>std::string</code> Wajib: Tidak
T	Merupakan nilai atribut numerik. Jenis: <code>double</code> Wajib: Tidak
SL	Merupakan array nilai atribut string. Jenis: <code>std::vector<std::string></code> Wajib: Tidak
SDM	Merupakan kamus kunci string dan nilai ganda. Jenis: <code>std::map<std::string, double></code> Wajib: Tidak

GetFleetRoleCredentialsRequest

Jenis data ini memberi server game akses terbatas ke AWS sumber daya Anda yang lain. Untuk informasi selengkapnya, lihat [Menyiapkan peran layanan IAM untuk Amazon GameLift Servers](#).

Properti	Deskripsi
RoleArn	Nama Sumber Daya Amazon (ARN) dari peran layanan yang memperluas akses terbatas ke sumber daya Anda. AWS Jenis: <code>std::string</code> Wajib: Tidak
RoleSessionName	Nama sesi peran yang dapat Anda gunakan untuk mengidentifikasi sesi secara unik. AWS Security Token Service AssumeRole Nama ini diekspos di log audit seperti yang ada di CloudTrail. Jenis: <code>std::string</code> Wajib: Tidak

AwsLongOutcome

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan. Jenis: <code>long</code> Wajib: Tidak
ResultWithOwnership	Hasil dari tindakan, dilemparkan sebagai rvalue, sehingga kode panggilan dapat mengambil kepemilikan objek. Jenis: <code>long&&</code> Wajib: Tidak
Berhasil	Apakah tindakan itu berhasil atau tidak.

Properti	Deskripsi
	Jenis: <code>bool</code>
	Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil.
	Jenis: the section called “GameLiftError”
	Wajib: Tidak

AwsStringOutcome

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan.
	Jenis: <code>std::string</code>
	Wajib: Tidak
ResultWithOwnership	Hasil dari tindakan, dilemparkan sebagai rvalue, sehingga kode panggilan dapat mengambil kepemilikan objek.
	Jenis: <code>long&&</code>
	Wajib: Tidak
Berhasil	Apakah tindakan itu berhasil atau tidak.
	Jenis: <code>bool</code>
	Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil.
	Jenis: the section called “GameLiftError”

Properti	Deskripsi
	Wajib: Tidak

DescribePlayerSessionsOutcome

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan. Jenis: the section called “DescribePlayerSessionsResult” Wajib: Tidak
ResultWithOwnership	Hasil dari tindakan, dilemparkan sebagai rvalue, sehingga kode panggilan dapat mengambil kepemilikan objek. Jenis: <code>Aws::GameLift::Server::Model::DescribePlayerSessionsResult&&</code> Wajib: Tidak
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: <code>bool</code> Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Jenis: the section called “GameLiftError” Wajib: Tidak

DescribePlayerSessionsResult

Kumpulan objek yang berisi properti untuk setiap sesi pemain yang cocok dengan permintaan.

Properti	Deskripsi
NextToken	Token yang menunjukkan awal halaman hasil sekuensia l berikutnya. Gunakan token yang dikembalikan dengan panggilan sebelumnya untuk operasi ini. Untuk memulai di awal set hasil, jangan tentukan nilai. Jika ID sesi pemain ditentukan, parameter ini diabaikan. Jenis: <code>std::string</code> Wajib: Ya
PlayerSessions	Jenis: <code>IList<the section called "PlayerSession"></code> Diperlukan:
ResultWithOwnership	Hasil dari tindakan, dilemparkan sebagai rvalue, sehingga kode panggilan dapat mengambil kepemilikan objek. Jenis: <code>std::string&&</code> Wajib: Tidak
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: <code>bool</code> Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Jenis: the section called "GameLiftError" Wajib: Tidak

GenericOutcome

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: bool Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Jenis: the section called “GameLiftError” Wajib: Tidak

GenericOutcomeCallable

Tipe data ini adalah hasil generik asinkron. Ini memiliki sifat-sifat berikut:

Properti	Deskripsi
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: bool Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Jenis: the section called “GameLiftError” Wajib: Tidak

PlayerSession

Tipe data ini mewakili sesi pemain yang Amazon GameLift Servers lolos ke server game. Untuk informasi selengkapnya, lihat [PlayerSession](#).

Properti	Deskripsi
CreationTime	Jenis: long Wajib: Tidak
FleetId	Jenis: std::string Wajib: Tidak
GameSessionId	Jenis: std::string Wajib: Tidak
IpAddress	Jenis: std::string Wajib: Tidak
PlayerData	Jenis: std::string Wajib: Tidak
PlayerId	Jenis: std::string Wajib: Tidak
PlayerSessionId	Jenis: std::string Wajib: Tidak
Port	Jenis: int Wajib: Tidak
Status	Status sesi pemain untuk mem-filter hasil. Ketika a PlayerSessionId atau PlayerId disediakan, maka tidak PlayerSessionStatusFilter berpengaruh pada respons. Jenis: A PlayerSessionStatus enum. Kemungkinan nilainya mencakup berikut ini: • AKTIF

Properti	Deskripsi
	• SELESAI • NOT_SET • DIPESAN • TIMEDOUT Wajib: Tidak
TerminationTime	Jenis: long Wajib: Tidak
DnsName	Jenis: std::string Wajib: Tidak

StartMatchBackfillOutcome

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan. Jenis: the section called “StartMatchBackfillResult” Wajib: Tidak
ResultWithOwnership	Hasil dari tindakan, dilemparkan sebagai rvalue, sehingga kode panggilan dapat mengambil kepemilikan objek. Jenis: StartMatchBackfillResult&& Wajib: Tidak
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: bool

Properti	Deskripsi
	Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Jenis: the section called “GameLiftError” Wajib: Tidak

StartMatchBackfillResult

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
TicketId	Pengenal unik untuk tiket perjudian. Jika tidak ada ID tiket yang ditentukan di sini, Amazon GameLift Servers akan menghasilkan satu dalam bentuk UUID. Gunakan pengenal ini untuk melacak status tiket isi ulang pertandingan dan mengambil hasil pertandingan. Jenis: <code>std::string</code> Wajib: Tidak

GetComputeCertificateOutcome

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan. Jenis: the section called “GetComputeCertificateResult” Wajib: Tidak

Properti	Deskripsi
ResultWithOwnership	Hasil dari tindakan, dilemparkan sebagai rvalue, sehingga kode panggilan dapat mengambil kepemilikan objek. Jenis: <code>Aws::GameLift::Server::Model::GetComputeCertificateResult&&</code> Wajib: Tidak
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: <code>bool</code> Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Jenis: the section called “GameLiftError” Wajib: Tidak

GetComputeCertificateResult

Jalur ke sertifikat TLS pada komputasi Anda dan nama host komputasi.

Properti	Deskripsi
CertificatePath	Jalur ke sertifikat TLS pada sumber daya komputasi Anda. Saat menggunakan armada Amazon GameLift Servers terkelola, jalur ini berisi: <code>certificate.pem</code> : Sertifikat pengguna akhir. Rantai sertifikat lengkap adalah kombinasi dari <code>certificateChain.pem</code> ditambahkan ke sertifikat ini. <code>certificateChain.pem</code> : Rantai sertifikat yang berisi sertifikat root dan sertifikat perantara. <code>rootCertificate.pem</code> : Sertifikat root.

Properti	Deskripsi
	<code>privateKey.pem</code> : Kunci pribadi untuk sertifikat pengguna akhir. Jenis: <code>std::string</code> Wajib: Tidak
ComputeName	Nama sumber daya komputasi Anda. Jenis: <code>std::string</code> Wajib: Tidak

GetFleetRoleCredentialsOutcome

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan. Jenis: the section called “GetFleetRoleCredentialsResult” Wajib: Tidak
ResultWithOwnership	Hasil dari tindakan, dilemparkan sebagai rvalue, sehingga kode panggilan dapat mengambil kepemilikan objek. Jenis: <code>Aws::GameLift::Server::Model::GetFleetRoleCredentialsResult</code> Wajib: Tidak
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: <code>bool</code> Wajib: Ya

Properti	Deskripsi
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Jenis: the section called “GameLiftError” Wajib: Tidak

GetFleetRoleCredentialsResult

Properti	Deskripsi
AccessKeyId	ID kunci akses untuk mengautentikasi dan menyediakan akses ke AWS sumber daya Anda. Jenis: <code>string</code> Wajib: Tidak
AssumedRoleId	ID pengguna yang menjadi milik peran layanan. Jenis: <code>string</code> Wajib: Tidak
AssumedRoleUserArn	Nama Sumber Daya Amazon (ARN) pengguna yang menjadi milik peran layanan. Jenis: <code>string</code> Wajib: Tidak
Kedaluwarsa	Jumlah waktu hingga kredensi sesi Anda kedaluwarsa. Jenis: <code>DateTime</code> Wajib: Tidak
SecretAccessKey	ID kunci akses rahasia untuk otentikasi. Jenis: <code>string</code>

Properti	Deskripsi
	Wajib: Tidak
SessionToken	Token untuk mengidentifikasi sesi aktif saat ini yang berinteraksi dengan AWS sumber daya Anda. Jenis: <code>string</code> Wajib: Tidak
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: <code>bool</code> Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Jenis: the section called “GameLiftError” Wajib: Tidak

Init SDKOutcome

 Note

InitSDKOutcomedikembalikan hanya ketika Anda membangun SDK dengan `std` flag. Jika Anda membangun dengan `nostd` bendera, maka [the section called “GenericOutcome”](#) dikembalikan sebagai gantinya.

Properti	Deskripsi
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: <code>bool</code> Wajib: Ya

Properti	Deskripsi
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Jenis: the section called “GameLiftError” Wajib: Tidak

GameLiftError

Properti	Deskripsi
ErrorType	Jenis kesalahan. Jenis: A GameLiftErrorType enum. Wajib: Tidak
ErrorMessage	Pesan kesalahan. Jenis: std::string Wajib: Tidak
ErrorName	Nama kesalahannya. Jenis: std::string Wajib: Tidak

Enum

Enum yang didefinisikan untuk SDK server untuk Amazon GameLift Servers (C++) didefinisikan sebagai berikut:

GameLiftErrorType

Nilai string menunjukkan jenis kesalahan. Nilai yang valid meliputi:

- BAD_REQUEST_EXCEPTION

- `GAMESESSION_ID_NOT_SET` — ID sesi permainan belum ditetapkan.
- `INTERNAL_SERVICE_EXCEPTION`
- `LOCAL_CONNECTION_FAILED` — Koneksi lokal gagal. Amazon GameLift Servers
- `NETWORK_NOT_INITIALIZED` — Jaringan belum diinisialisasi.
- `SERVICE_CALL_FAILED` - Panggilan ke layanan telah gagal. AWS
- `WEBSOCKET_CONNECT_FAILURE`
- `WEBSOCKET_CONNECT_FAILURE_FORBIDDEN`
- `WEBSOCKET_CONNECT_FAILURE_INVALID_URL`
- `WEBSOCKET_CONNECT_FAILURE_TIMEOUT`
- `ALREADY_INITIALIZED` — Amazon GameLift Servers Server atau Klien telah diinisialisasi dengan `Initialize ()`.
- `FLEET_MISMATCH` — Armada target tidak cocok dengan armada `GameSession` atau `PlayerSession`.
- `GAMELIFT_CLIENT_NOT_INITIALIZED` — Klien belum diinisialisasi. Amazon GameLift Servers
- `GAMELIFT_SERVER_NOT_INITIALIZED` — Server belum diinisialisasi. Amazon GameLift Servers
- `GAME_SESSION_ENDED_FAILED` — SDK server untuk tidak Amazon GameLift Servers dapat menghubungi layanan untuk melaporkan sesi permainan berakhir.
- `GAME_SESSION_NOT_READY` — Sesi Game Server tidak diaktifkan. Amazon GameLift Servers
- `GAME_SESSION_READY_FAILED` — SDK server untuk tidak Amazon GameLift Servers dapat menghubungi layanan untuk melaporkan sesi permainan sudah siap.
- `INITIALIZATION_MISMATCH` — Metode klien dipanggil setelah `Server: :Initialize ()`, atau sebaliknya.
- `NOT_INITIALIZED` — Amazon GameLift Servers Server atau Klien belum diinisialisasi dengan `Initialize ()`.
- `NO_TARGET_ALIASID_SET` — AliaID target belum ditetapkan.
- `NO_TARGET_FLEET_SET` — Armada target belum ditetapkan.
- `PROCESS_ENDING_FAILED` — SDK server untuk tidak Amazon GameLift Servers dapat menghubungi layanan untuk melaporkan proses berakhir.
- `PROCESS_NOT_ACTIVE` — Proses server belum aktif, tidak terikat pada `GameSession`, dan tidak dapat menerima atau memproses. `PlayerSessions`

- `PROCESS_NOT_READY` — Proses server belum siap untuk diaktifkan.
- `PROCESS_READY_FAILED` — SDK server untuk tidak Amazon GameLift Servers dapat menghubungi layanan untuk melaporkan proses siap.
- `SDK_VERSION_DETECTION_FAILED` — Deteksi versi SDK gagal.
- `STX_CALL_FAILED` — Panggilan ke komponen backend XStx server telah gagal.
- `STX_INITIALIZATION_FAILED` — Komponen backend server gagal XStx menginisialisasi.
- `UNEXPECTED_PLAYER_SESSION` — Sesi pemain yang tidak terdaftar ditemui oleh server.
- `WEBSOCKET_CONNECT_FAILURE`
- `WEBSOCKET_CONNECT_FAILURE_FORBIDDEN`
- `WEBSOCKET_CONNECT_FAILURE_INVALID_URL`
- `WEBSOCKET_CONNECT_FAILURE_TIMEOUT`
- `WEBSOCKET_RETRIABLE_SEND_MESSAGE_FAILURE` - Kegagalan yang dapat diambil untuk mengirim pesan ke Layanan. GameLift WebSocket
- `WEBSOCKET_SEND_MESSAGE_FAILURE` — Kegagalan untuk mengirim pesan ke Layanan. GameLift WebSocket
- `MATCH_BACKFILL_REQUEST_VALIDATION` — Validasi permintaan gagal.
- `PLAYER_SESSION_REQUEST_VALIDATION` — Validasi permintaan gagal.

PlayerSessionCreationPolicy

Nilai string yang menunjukkan apakah sesi game menerima pemain baru. Nilai yang valid meliputi:

- `ACCEPT_ALL` — Menerima semua sesi pemain baru.
- `DENY_ALL` — Menolak semua sesi pemain baru.
- `NOT_SET` — Sesi permainan tidak diatur untuk menerima atau menolak sesi pemain baru.

[Server C++ SDK 5.x untuk Amazon GameLift Servers -- Tipe data](#)

Topik

- [GetSdkVersion\(\)](#)
- [InitSDK\(\)](#)
- [InitSDK\(\)](#)
- [ProcessReady\(\)](#)

- [ProcessReadyAsync\(\)](#)
- [ProcessEnding\(\)](#)
- [ActivateGameSession\(\)](#)
- [UpdatePlayerSessionCreationPolicy\(\)](#)
- [GetGameSessionId\(\)](#)
- [GetTerminationTime\(\)](#)
- [AcceptPlayerSession\(\)](#)
- [RemovePlayerSession\(\)](#)
- [DescribePlayerSessions\(\)](#)
- [StartMatchBackfill\(\)](#)
- [StopMatchBackfill\(\)](#)
- [GetComputeCertificate\(\)](#)
- [GetFleetRoleCredentials\(\)](#)
- [Hancurkan \(\)](#)

GetSdkVersion()

Mengembalikan nomor versi SDK saat ini yang dibangun ke dalam proses server.

Sintaks

```
Aws::GameLift::AwsStringOutcome Server::GetSdkVersion();
```

Nilai yang ditampilkan

Jika berhasil, ini mengembalikan versi SDK saat ini sebagai objek [the section called “AwsStringOutcome”](#). Objek yang dikembalikan menyertakan nomor versi (contoh 5.0.0). Jika tidak berhasil, ini mengembalikan pesan kesalahan.

Contoh

```
Aws::GameLift::AwsStringOutcome SdkVersionOutcome =  
    Aws::GameLift::Server::GetSdkVersion();
```

InitSDK()

Menginisialisasi Amazon GameLift Servers SDK. Panggil metode ini saat diluncurkan sebelum langkah inisialisasi lain yang terkait Amazon GameLift Servers dengan. Tindakan ini membaca parameter server dari lingkungan host untuk mengatur komunikasi antara proses server game dan Amazon GameLift Servers layanan.

Jika build server game akan dikerahkan tanpa Amazon GameLift Servers Agen ke armada atau armada kontainer Amazon GameLift Servers Anywhere, panggil [InitSDK\(\)](#) dan tentukan satu set parameter server.

Sintaks

```
Server::InitSDKOutcome Server::initSdkOutcome = InitSDK();
```

Nilai yang ditampilkan

Mengembalikan [the section called "Init SDKOutcome"](#) objek yang menunjukkan apakah proses server siap untuk memanggil [ProcessReady\(\)](#).

Contoh

```
//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()

Menginisialisasi Amazon GameLift Servers SDK. Panggil metode ini saat diluncurkan sebelum langkah inisialisasi lain yang terkait Amazon GameLift Servers dengan. Tindakan ini membutuhkan seperangkat parameter server untuk mengatur komunikasi antara proses server game dan Amazon GameLift Servers layanan.

Jika build server game akan dikerahkan ke armada Amazon GameLift Servers terkelola atau ke EC2 armada Amazon GameLift Servers Anywhere atau armada kontainer dengan Amazon GameLift Servers Agen, hubungi [InitSDK\(\)](#) tanpa parameter server.

Sintaks

```
Server::InitSDKOutcome Server::initSdkOutcome = InitSDK(serverParameters);
```

Parameter

ServerParameters

Untuk menginisialisasi server game pada armada Amazon GameLift Servers Anywhere, buat `ServerParameters` objek dengan informasi berikut:

- URL yang WebSocket digunakan untuk terhubung ke server game Anda.
- ID dari proses yang digunakan untuk meng-host server game Anda.
- ID komputasi yang menghosting proses server game Anda.
- ID Amazon GameLift Servers armada yang berisi komputasi Amazon GameLift Servers Anywhere Anda.
- Token otorisasi yang dihasilkan oleh Amazon GameLift Servers operasi.

Nilai yang dikembalikan

Mengembalikan [the section called "Init SDKOutcome"](#) objek yang menunjukkan apakah proses server siap untuk memanggil [ProcessReady\(\)](#).

Note

Jika panggilan gagal untuk build game yang diterapkan ke `InitSDK()` armada Anywhere, periksa `ServerSdkVersion` parameter yang digunakan saat membuat sumber daya build. Anda harus secara eksplisit menetapkan nilai ini ke versi SDK server yang digunakan. Nilai default untuk parameter ini adalah 4.x, yang tidak kompatibel. Untuk mengatasi masalah ini, buat build baru dan terapkan ke armada baru.

Contoh

Amazon GameLift ServersContoh di mana saja

```
//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()

Memberitahu Amazon GameLift Servers bahwa proses server siap untuk meng-host sesi permainan. Panggil metode ini setelah memanggil [InitSDK\(\)](#). Metode ini harus dipanggil hanya satu kali per proses.

Sintaks

```
GenericOutcome ProcessReady(const Aws::GameLift::Server::ProcessParameters
&processParameters);
```

Parameter

processParameters

Sebuah objek [ProcessParameters](#) yang mengomunikasikan informasi berikut tentang proses server:

- Nama metode callback yang diimplementasikan dalam kode server game yang dipanggil Amazon GameLift Servers layanan untuk berkomunikasi dengan proses server.
- Nomor port yang didengarkan oleh proses server.
- Jalur ke file khusus sesi game apa pun yang Amazon GameLift Servers ingin Anda tangkap dan simpan.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Contoh ini menggambarkan panggilan [ProcessReady\(\)](#) dan mendelegasi implementasi fungsi.

```
// 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()

Memberitahu Amazon GameLift Servers layanan bahwa proses server siap untuk meng-host sesi permainan. Metode ini harus dipanggil setelah proses server siap untuk meng-host sesi permainan. Parameter menentukan nama fungsi callback Amazon GameLift Servers untuk dipanggil dalam keadaan tertentu. Kode server game harus menerapkan fungsi-fungsi ini.

Ini adalah panggilan asinkron. Untuk membuat panggilan sinkron, gunakan [ProcessReady\(\)](#). Lihat [Inisialisasi proses server](#) untuk detail selengkapnya.

Sintaks

```
GenericOutcomeCallable ProcessReadyAsync(  
    const Aws::GameLift::Server::ProcessParameters &processParameters);
```

Parameter

processParameters

Sebuah objek [ProcessParameters](#) yang mengomunikasikan informasi berikut tentang proses server:

- Nama metode callback yang diimplementasikan dalam kode server game yang dipanggil Amazon GameLift Servers layanan untuk berkomunikasi dengan proses server.
- Nomor port yang didengarkan oleh proses server.
- Jalur ke file khusus sesi game apa pun yang Amazon GameLift Servers ingin Anda tangkap dan simpan.

Wajib: Ya

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

```
// 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()

Memberitahu Amazon GameLift Servers bahwa proses server berakhir. Panggil metode ini setelah semua tugas pembersihan lainnya (termasuk mematikan sesi permainan aktif) dan sebelum mengakhiri proses. Tergantung pada hasil `ProcessEnding()`, proses keluar dengan sukses (0) atau kesalahan (-1) dan menghasilkan peristiwa armada. Jika proses berakhir dengan kesalahan, peristiwa armada yang dihasilkan adalah `SERVER_PROCESS_TERMINATED_UNHEALTHY`.

Sintaks

```
Aws::GameLift::GenericOutcome processEndingOutcome =  
    Aws::GameLift::Server::ProcessEnding();
```

Nilai yang ditampilkan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Contoh ini memanggil `ProcessEnding()` dan `Destroy()` sebelum menghentikan proses server dengan kode keluar sukses atau kesalahan.

```
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()

Memberitahu Amazon GameLift Servers bahwa proses server telah mengaktifkan sesi permainan dan sekarang siap menerima koneksi pemain. Tindakan ini harus dipanggil sebagai bagian dari fungsi `onStartGameSession()` callback, setelah semua inisialisasi sesi game.

Sintaks

```
Aws::GameLift::GenericOutcome activateGameSessionOutcome =  
    Aws::GameLift::Server::ActivateGameSession();
```

Nilai yang ditampilkan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Contoh ini menunjukkan `ActivateGameSession()` dipanggil sebagai bagian dari fungsi `onStartGameSession()` delegasi.

```
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()

Memperbarui kemampuan sesi game saat ini untuk menerima sesi pemain baru. Sesi game dapat diatur untuk menerima atau menolak semua sesi pemain baru.

Sintaks

```
GenericOutcome
UpdatePlayerSessionCreationPolicy(Aws::GameLift::Model::PlayerSessionCreationPolicy
newPlayerSessionPolicy);
```

Parameter

playerCreationSessionKebijakan

Jenis: nilai `PlayerSessionCreationPolicy` [enum](#).

Wajib: Ya

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Contoh ini menetapkan kebijakan bergabung sesi game saat ini untuk menerima semua pemain.

```
Aws::GameLift::GenericOutcome outcome =  
  
    Aws::GameLift::Server::UpdatePlayerSessionCreationPolicy(Aws::GameLift::Model::PlayerSessionCr
```

GetGameSessionId()

Mengambil ID sesi permainan yang dihosting oleh proses server aktif.

Untuk proses idle yang tidak diaktifkan dengan sesi game, panggilan akan menampilkan file. [the section called “GameLiftError”](#)

Sintaks

```
AwsStringOutcome GetGameSessionId()
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai yang dikembalikan

Jika berhasil, ini mengembalikan ID sesi game sebagai objek [the section called “AwsStringOutcome”](#).
Jika tidak berhasil, ini mengembalikan pesan kesalahan.

Untuk proses idle yang tidak diaktifkan dengan sesi game, panggilan mengembalikan Success = True dan GameSessionId = "".

Contoh

```
Aws::GameLift::AwsStringOutcome sessionIdOutcome =  
    Aws::GameLift::Server::GetGameSessionId();
```

GetTerminationTime()

Mengembalikan waktu yang merupakan jadwal proses server akan ditutup, jika waktu penghentian tersedia. Proses server mengambil tindakan setelah menerima onProcessTerminate() panggilan

balik dari Amazon GameLift Servers. Amazon GameLift Servers memanggil `onProcessTerminate()` untuk alasan berikut:

- Ketika proses server telah melaporkan kesehatan yang buruk atau belum Amazon GameLift Servers merespons.
- Saat mengakhiri instance selama acara scale-down.
- Ketika sebuah instance dihentikan karena gangguan [spot-instance](#).

Sintaks

```
AwsDateTimeOutcome GetTerminationTime()
```

Nilai yang ditampilkan

Jika berhasil, ini mengembalikan waktu penghentian sebagai objek `AwsDateTimeOutcome`. Nilainya adalah waktu penghentian, dinyatakan dalam kutu yang telah berlalu sejak. `000100:00:00` Misalnya, nilai waktu tanggal `2020-09-13 12:26:40 -000Z` sama dengan `637355968000000000` kutu. Jika tidak ada waktu penghentian tersedia, pesan kesalahan ditampilkan.

Jika proses belum menerima `ProcessParameters`. `OnProcessTerminate()` callback, pesan kesalahan dikembalikan. Untuk informasi selengkapnya tentang mematikan proses server, lihat [Menanggapi notifikasi shutdown proses server](#).

Contoh

```
Aws::GameLift::AwsLongOutcome TermTimeOutcome =  
    Aws::GameLift::Server::GetTerminationTime();
```

AcceptPlayerSession()

Memberitahu Amazon GameLift Servers bahwa pemain dengan ID sesi pemain tertentu telah terhubung ke proses server dan membutuhkan validasi. Amazon GameLift Servers memverifikasi bahwa ID sesi pemain valid. Setelah sesi pemain divalidasi, Amazon GameLift Servers mengubah status slot pemain dari `RESERVED` menjadi `AKTIF`.

Sintaks

```
GenericOutcome AcceptPlayerSession(String playerId)
```

Parameter

playerSessionId

ID unik yang dikeluarkan oleh Amazon GameLift Servers saat sesi pemain baru dibuat.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Contoh ini menangani permintaan koneksi yang mencakup memvalidasi dan menolak sesi pemain yang tidak valid. 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());
    }
}
```

RemovePlayerSession()

Memberitahu Amazon GameLift Servers bahwa pemain telah terputus dari proses server. Sebagai tanggapan, Amazon GameLift Servers ubah slot pemain menjadi tersedia.

Sintaks

```
GenericOutcome RemovePlayerSession(String playerSessionId)
```

Parameter

playerSessionId

ID unik yang dikeluarkan oleh Amazon GameLift Servers saat sesi pemain baru dibuat.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

```
Aws::GameLift::GenericOutcome disconnectOutcome =  
    Aws::GameLift::Server::RemovePlayerSession(playerSessionId);
```

DescribePlayerSessions()

Mengambil data sesi pemain yang mencakup pengaturan, metadata sesi, dan data pemain. Gunakan metode ini untuk mendapatkan informasi tentang hal-hal berikut:

- Sesi pemain tunggal
- Semua sesi pemain dalam sesi permainan
- Semua sesi pemain yang terkait dengan ID pemain tunggal

Sintaks

```
DescribePlayerSessionsOutcome DescribePlayerSessions(DescribePlayerSessionsRequest  
    describePlayerSessionsRequest)
```

Parameter

[DescribePlayerSessionsRequest](#)

[the section called “DescribePlayerSessionsRequest”](#) Objek yang menggambarkan sesi pemain mana yang akan diambil.

Nilai yang dikembalikan

Jika berhasil, ini mengembalikan objek [the section called “DescribePlayerSessionsOutcome”](#) yang berisi satu set objek sesi pemain yang sesuai dengan parameter permintaan.

Contoh

Contoh ini meminta semua sesi pemain yang terhubung secara aktif ke sesi permainan tertentu. Dengan menghilangkan NextToken dan mengatur nilai Batas ke 10, Amazon GameLift Servers mengembalikan catatan sesi 10 pemain pertama yang cocok dengan permintaan.

```
// 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()

Mengirim permintaan untuk menemukan pemain baru untuk slot terbuka dalam sesi permainan yang dibuat dengan FlexMatch. Untuk informasi selengkapnya, lihat [fitur FlexMatch isi ulang](#).

Tindakan ini asinkron. Jika pemain baru dicocokkan, kirimkan data Amazon GameLift Servers mak comblang yang diperbarui menggunakan fungsi panggilan balik. `OnUpdateGameSession()`

Proses server hanya dapat melakukan satu permintaan backfill match yang aktif dalam satu waktu. Untuk mengirim permintaan baru, panggil [StopMatchBackfill\(\)](#) terlebih dahulu untuk membatalkan permintaan asli.

Sintaks

```
StartMatchBackfillOutcome StartMatchBackfill (StartMatchBackfillRequest
    startBackfillRequest);
```

Parameter

[StartMatchBackfillRequest](#)

StartMatchBackfillRequest Objek yang mengkomunikasikan informasi berikut:

- ID tiket untuk ditetapkan ke permintaan backfill. Informasi ini opsional; jika tidak ada ID yang diberikan, Amazon GameLift Servers akan menghasilkan satu.
- Matchmaker untuk dikirim permintaan. ARN konfigurasi penuh diperlukan. Nilai ini ada dalam data mak comblang sesi permainan.
- ID sesi permainan untuk mengisi ulang.
- Data perjodohan yang tersedia untuk pemain sesi permainan saat ini.

Nilai yang dikembalikan

Mengembalikan [the section called “StartMatchBackfillOutcome”](#) objek dengan ID tiket isi ulang kecocokan, atau kegagalan dengan pesan kesalahan.

Contoh

```
// 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()

Membatalkan permintaan pengisian ulang pertandingan yang aktif. Untuk informasi selengkapnya, lihat [fitur FlexMatch isi ulang](#).

Sintaks

```
GenericOutcome StopMatchBackfill (StopMatchBackfillRequest stopBackfillRequest);
```

Parameter

[StopMatchBackfillRequest](#)

StopMatchBackfillRequest Objek yang mengidentifikasi tiket perijodohan untuk dibatalkan:

- ID tiket yang ditetapkan untuk permintaan pengisian ulang.
- Mak comblang permintaan isi ulang dikirim ke.
- Sesi permainan yang terkait dengan permintaan isi ulang.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

```
// 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()

Mengambil jalur ke sertifikat TLS yang digunakan untuk mengenkripsi koneksi jaringan antara sumber daya komputasi Amazon GameLift Servers Anywhere Anda dan Amazon GameLift Servers Anda dapat menggunakan jalur sertifikat saat mendaftarkan perangkat komputasi ke armada Amazon GameLift Servers Anywhere. Untuk informasi selengkapnya, lihat [RegisterCompute](#).

Sintaks

```
GetComputeCertificateOutcome Server::GetComputeCertificate()
```

Nilai yang ditampilkan

Mengembalikan [the section called “GetComputeCertificateOutcome”](#).

Contoh

```
Aws::GameLift::GetComputeCertificateOutcome certificate =  
    Aws::GameLift::Server::GetComputeCertificate();
```

GetFleetRoleCredentials()

Mengambil kredensial peran IAM yang mengizinkan Amazon GameLift Servers untuk berinteraksi dengan orang lain. Layanan AWS Untuk informasi selengkapnya, lihat [Berkomunikasi dengan AWS sumber daya lain dari armada Anda](#).

Sintaks

```
GetFleetRoleCredentialsOutcome GetFleetRoleCredentials(GetFleetRoleCredentialsRequest  
    request);
```

Parameter

[GetFleetRoleCredentialsRequest](#)

Nilai yang dikembalikan

Mengembalikan objek [the section called “GetFleetRoleCredentialsOutcome”](#).

Contoh

```
// 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);
```

Contoh ini menunjukkan penggunaan RoleSessionName nilai opsional untuk menetapkan nama ke sesi kredensial untuk tujuan audit. Jika Anda tidak memberikan nama sesi peran, nilai default ***"[fleet-id]-[host-id]"*** digunakan.

```
// 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);
```

Hancurkan ()

Membebaskan SDK server Amazon GameLift Servers game dari memori. Sebagai praktik terbaik, hubungi metode ini setelah ProcessEnding() dan sebelum mengakhiri proses. Jika Anda menggunakan armada Anywhere dan Anda tidak menghentikan proses server setelah setiap sesi game, panggil Destroy() dan kemudian InitSDK() untuk menginisialisasi ulang sebelum memberi tahu Amazon GameLift Servers bahwa proses tersebut siap untuk menjadi tuan rumah sesi game. ProcessReady()

Sintaks

```
GenericOutcome Aws::GameLift::Server::Destroy();
```

Parameter

Tidak ada parameter.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

```
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);  
}
```

C# server SDK 5.x untuk Amazon GameLift Servers -- Tindakan

Gunakan referensi SDK 5.x server untuk mengintegrasikan game multipemain Anda untuk hosting. Amazon GameLift Servers Untuk panduan tentang proses integrasi, lihat [Tambahkan Amazon GameLift Servers ke server game Anda](#). Jika Anda menggunakan Amazon GameLift Servers plugin untuk Unity, lihat juga [Amazon GameLift Serversplugin untuk Unity \(server SDK 5.x\)](#).

C# server SDK 5.x untuk Amazon GameLift Servers - Tipe data

Gunakan referensi SDK 5.x server Amazon GameLift Servers C # untuk mengintegrasikan game multipemain Anda untuk hosting. Amazon GameLift Servers Untuk panduan tentang proses integrasi, lihat [Tambahkan Amazon GameLift Servers ke server game Anda](#). Jika Anda menggunakan Amazon GameLift Servers plugin untuk Unity, lihat juga [Amazon GameLift Serversplugin untuk Unity \(server SDK 5.x\)](#).

[C# server SDK 5.x untuk Amazon GameLift Servers -- Tindakan](#)

Jenis Data

- [LogParameters](#)
- [ProcessParameters](#)

- [UpdateGameSession](#)
- [GameSession](#)
- [ServerParameters](#)
- [StartMatchBackfillRequest](#)
- [Pemain](#)
- [DescribePlayerSessionsRequest](#)
- [StopMatchBackfillRequest](#)
- [GetFleetRoleCredentialsRequest](#)
- [AttributeValue](#)
- [AwsStringOutcome](#)
- [GenericOutcome](#)
- [DescribePlayerSessionsOutcome](#)
- [DescribePlayerSessionsResult](#)
- [PlayerSession](#)
- [StartMatchBackfillOutcome](#)
- [StartMatchBackfillResult](#)
- [GetComputeCertificateOutcome](#)
- [GetComputeCertificateResult](#)
- [GetFleetRoleCredentialsOutcome](#)
- [GetFleetRoleCredentialsResult](#)
- [AwsDateTimeOutcome](#)
- [GameLiftError](#)
- [Enum](#)

LogParameters

Gunakan tipe data ini untuk mengidentifikasi file mana yang dihasilkan selama sesi permainan yang Anda inginkan untuk diunggah oleh server game Amazon GameLift Servers setelah sesi permainan berakhir. Server game berkomunikasi LogParameters to Amazon GameLift Servers dalam [ProcessReady\(\)](#) panggilan.

Sifat-sifat	Deskripsi

LogPaths	Daftar jalur direktori ke file log server game yang ingin Anda simpan Amazon GameLift Servers untuk akses future. Proses server menghasilkan file-file ini selama setiap sesi permainan. Anda menentukan jalur dan nama file di server game Anda dan menyimpannya di direktori build game root. Jalur log harus absolut. Misalnya, jika build game Anda menyimpan log sesi game di jalur seperti <code>MyGame\sessionLogs\</code> , maka jalurnya akan berada <code>c:\game\MyGame\sessionLogs</code> di instance Windows. Jenis: <code>List<String></code> Wajib: Tidak

ProcessParameters

Tipe data ini berisi kumpulan parameter yang dikirim ke Amazon GameLift Servers dalam [ProcessReady\(\)](#) panggilan.

Sifat-sifat	Deskripsi
LogParameters	Objek dengan daftar jalur direktori ke file log sesi game. Jenis: <code>Aws::GameLift::Server:: LogParameters</code> Wajib: Ya
OnHealthCheck	Nama fungsi callback yang Amazon GameLift Servers memanggil untuk meminta laporan status kesehatan dari proses server. Amazon GameLift Servers memanggil fungsi ini setiap 60 detik. Setelah memanggil fungsi ini Amazon

GameLift Servers menunggu 60 detik untuk respons, jika tidak ada yang diterima, Amazon GameLift Servers mencatat proses server sebagai tidak sehat.

Jenis: `void OnHealthCheckDelegate()`

Wajib: Ya

OnProcessTerminate

Nama fungsi callback yang Amazon GameLift Servers memanggil untuk memaksa proses server dimatikan. Setelah memanggil fungsi ini, Amazon GameLift Servers tunggu lima menit hingga proses server dimatikan dan merespons dengan [ProcessEnding\(\)](#) panggilan sebelum mematikan proses server.

Jenis: `void OnProcessTerminate Delegate()`

Wajib: Ya

OnStartGameSession

Nama fungsi callback yang Amazon GameLift Servers memanggil untuk mengaktifkan sesi permainan baru. Amazon GameLift Servers memanggil fungsi ini dalam menanggapi permintaan klien [CreateGameSession](#). Fungsi callback mengambil [GameSession](#) objek.

Jenis: `void OnStartGameSession Delegate( GameSession )`

Wajib: Ya

OnUpdateGameSession	Nama fungsi callback yang Amazon GameLift Servers memanggil untuk meneruskan objek sesi permainan yang diperbarui ke proses server. Amazon GameLift Servers memanggil fungsi ini ketika permintaan pengisian ulang kecocokan telah diproses untuk menyediakan data mak comblang yang diperbarui. Ini melewati GameSession objek, pembaruan status (updateReason), dan ID tiket isi ulang pertandingan. Jenis: void OnUpdateGameSessionDelegate () UpdateGameSession Wajib: Tidak
Port	Nomor port yang didengarkan oleh proses server untuk koneksi pemain baru. Nilai harus masuk ke dalam kisaran port yang dikonfigurasi untuk setiap armada yang men-deploy build server game ini. Nomor port ini termasuk dalam sesi game dan objek sesi pemain, yang digunakan sesi game saat menghubungkan ke proses server. Jenis: Integer Wajib: Ya

UpdateGameSession

Informasi yang diperbarui untuk objek sesi permainan, termasuk alasan bahwa sesi permainan diperbarui. Jika pembaruan terkait dengan tindakan pengisian ulang kecocokan, tipe data ini menyertakan ID tiket isi ulang.

Properti	Deskripsi
GameSession	Sebuah objek GameSession. GameSession Objek berisi properti yang menggambarkan sesi permainan. Jenis: GameSession GameSession() Wajib: Ya
UpdateReason	Alasan bahwa sesi permainan sedang diperbarui. Jenis: UpdateReason UpdateReason() Wajib: Ya
BackfillTicketId	ID tiket isi ulang yang mencoba memperbarui sesi permainan. Jenis: String Wajib: Ya

GameSession

Detail sesi permainan.

Properti	Deskripsi
GameSessionId	Pengenal unik untuk sesi permainan. Sesi permainan ARN memiliki format berikut: arn:aws:gamelift:<region>::gamesession/<fleet ID>/<custom ID string or idempotency token> Jenis: String Wajib: Tidak

Properti	Deskripsi
Nama	Label deskriptif dari sesi permainan. Jenis: String Wajib: Tidak
FleetId	Pengenal unik untuk armada tempat sesi permainan berjalan. Jenis: String Wajib: Tidak
MaximumPlayerSessionCount	Jumlah maksimum koneksi pemain ke sesi permainan. Jenis: Integer Wajib: Tidak
Port	Nomor port untuk sesi permainan. Untuk terhubung ke server Amazon GameLift Servers game, aplikasi memerlukan alamat IP dan nomor port. Jenis: Integer Wajib: Tidak
IpAddress	Alamat IP dari sesi game. Untuk terhubung ke server Amazon GameLift Servers game, aplikasi memerlukan alamat IP dan nomor port. Jenis: String Wajib: Tidak

Properti	Deskripsi
GameSessionData	Set properti sesi game khusus, diformat sebagai nilai string tunggal. Jenis: <code>String</code> Wajib: Tidak
MatchmakerData	Informasi tentang proses perjodohan yang digunakan untuk membuat sesi permainan, dalam sintaks JSON, diformat sebagai string. Selain konfigurasi perjodohan yang digunakan , ini berisi data tentang semua pemain yang ditugaskan untuk pertandingan, termasuk atribut pemain dan tugas tim. Jenis: <code>String</code> Wajib: Tidak
GameProperties	Satu set properti kustom untuk sesi permainan, diformat sebagai pasangan key:value. Properti ini diteruskan dengan permintaan untuk memulai sesi permainan baru. Jenis: <code>Dictionary<string, string></code> Wajib: Tidak

Properti	Deskripsi
DnsName	Pengenalan DNS yang ditetapkan ke instance yang menjalankan sesi permainan. Nilai memiliki format berikut: - Armada yang mendukung TLS:<unique identifier>.<region identifier>.amazongamelift.com . - Non-TLS-enabled armada:ec2-<unique identifier>.compute.amazonaws.com . Saat menghubungkan ke sesi permainan yang berjalan pada armada yang mendukung TLS, Anda harus menggunakan nama DNS, bukan alamat IP. Jenis: String Wajib: Tidak

ServerParameters

Informasi yang digunakan untuk menjaga koneksi antara server Amazon GameLift Servers Anywhere dan Amazon GameLift Servers layanan. Informasi ini digunakan saat meluncurkan proses server baru dengan [InitSDK\(\)](#). Untuk server yang dihosting pada EC2 instance Amazon GameLift Servers terkelola, gunakan objek kosong.

Properti	Deskripsi
WebSocketUrl	Yang <code>GameLiftServerSdkEndpoint</code> kembali ketika Anda <code>RegisterCompute</code> sebagai bagian dari Amazon GameLift Servers Anywhere. Jenis: String

Properti	Deskripsi
	Wajib: Ya
ProcessId	Pengenal unik yang terdaftar pada proses server yang menghosting game Anda. Jenis: String Wajib: Ya
HostId	Pengidentifikasi unik untuk host dengan server memproses hosting game Anda. HostID adalah yang digunakan saat Anda mendaftarkan ComputeName komputasi Anda. Untuk informasi lebih lanjut lihat, RegisterCompute Jenis: String Wajib: Ya
FleetId	ID armada armada tempat komputasi terdaftar . Untuk informasi selengkapnya, lihat RegisterCompute. Jenis: String Wajib: Ya
AuthToken	Token otentikasi yang dihasilkan oleh Amazon GameLift Servers itu mengotentikasi server Anda. Amazon GameLift Servers Untuk informasi selengkapnya, lihat GetComputeAuthToken. Jenis: String Wajib: Ya

StartMatchBackfillRequest

Informasi yang digunakan untuk membuat permintaan pengisian ulang perjodohan. Server game mengkomunikasikan informasi ini Amazon GameLift Servers dalam [StartMatchBackfill\(\)](#) panggilan.

Properti	Deskripsi
GameSessionArn	Pengidentifikasi sesi permainan yang unik. Operasi API GetGameSessionId mengembalikan pengenal dalam format ARN. Jenis: String Wajib: Ya
MatchmakingConfigurationArn	Pengidentifikasi unik, dalam bentuk ARN, untuk digunakan mak comblang untuk permintaan ini. ARN mak comblang untuk sesi permainan asli ada di objek sesi permainan di properti data mak comblang. Pelajari selengkapnya tentang data matchmaker di Bekerja dengan data matchmaker. Jenis: String Wajib: Ya
Pemain	Satu set data yang mewakili semua pemain yang saat ini dalam sesi permainan. Matchmaker menggunakan informasi ini untuk mencari pemain baru yang cocok untuk pemain saat ini. Jenis: List<Player> Wajib: Ya
TicketId	Pengenal unik untuk tiket permintaan pencocokan atau pencocokan isi ulang. Jika Anda tidak memberikan nilai, Amazon GameLift

Properti	Deskripsi
	Servers hasilkan satu. Gunakan pengidentifikasi ini untuk melacak status tiket backfill match atau membatalkan permintaan jika diperlukan. Jenis: String Wajib: Tidak

Pemain

Merupakan pemain dalam perjodohan. Ketika permintaan perjodohan dimulai, pemain memiliki ID pemain, atribut, dan mungkin data latensi. Amazon GameLift Servers menambahkan informasi tim setelah pertandingan dibuat.

Properti	Deskripsi
LatencyInMS	Satu set nilai yang dinyatakan dalam milidetik, yang menunjukkan jumlah latensi yang dialami pemain saat terhubung ke suatu lokasi. Jika properti ini digunakan, pemain hanya cocok untuk lokasi yang terdaftar. Jika mak comblang memiliki aturan yang mengevaluasi latensi pemain, pemain harus melaporkan latensi untuk dicocokkan. Jenis: Dictionary<string, int> Wajib: Tidak
PlayerAttributes	Kumpulan pasangan kunci:value yang berisi informasi pemain untuk digunakan dalam perjodohan. Kunci atribut pemain harus cocok dengan yang PlayerAttributes digunakan dalam set aturan perjodohan.

Properti	Deskripsi
	Untuk informasi selengkapnya tentang atribut pemain, lihat AttributeValue. Jenis: Dictionary<string, Attribute Value Wajib: Tidak
PlayerId	Pengenal unik untuk pemain. Jenis: String Wajib: Tidak
Tim	Nama tim yang ditugaskan pemain dalam pertandingan. Anda menentukan nama tim dalam set aturan perjodohan. Jenis: String Wajib: Tidak

DescribePlayerSessionsRequest

Jenis data ini digunakan untuk menentukan sesi pemain untuk diambil. Ini dapat digunakan dalam beberapa cara: (1) menyediakan PlayerSessionId untuk meminta sesi pemain tertentu; (2) menyediakan GameSessionId untuk meminta semua sesi pemain dalam sesi permainan yang ditentukan; atau (3) menyediakan PlayerId untuk meminta semua sesi pemain untuk pemain yang ditentukan. Untuk koleksi sesi pemain yang besar, gunakan parameter pemberian nomor halaman untuk mengambil hasil dalam halaman berurutan.

Properti	Deskripsi
GameSessionId	Pengidentifikasi sesi permainan yang unik. Gunakan parameter ini untuk meminta semua sesi pemain untuk sesi game yang ditentukan. Format ID sesi game adalah sebagai

Properti	Deskripsi
	berikut: <code>arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string></code>. Nilai <code><ID string></code> adalah string ID kustom (jika salah satu ditentukan saat sesi game dibuat) atau string yang dihasilkan. Jenis: <code>String</code> Wajib: Tidak
<code>PlayerSessionId</code>	Pengenal unik untuk sesi pemain. Jenis: <code>String</code> Wajib: Tidak
<code>PlayerId</code>	Pengenal unik untuk pemain. Lihat Hasilkan pemain IDs. Jenis: <code>String</code> Wajib: Tidak

Properti	Deskripsi
PlayerSessionStatusFilter	Status sesi pemain untuk memfilter hasil. Kemungkinan status sesi pemain meliputi: • RESERVED — Permintaan sesi pemain telah diterima, tetapi pemain belum terhubung ke proses server and/or telah divalidasi. • ACTIVE — Pemain telah divalidasi oleh proses server dan saat ini terhubung. • COMPLETED — Sambungan pemain telah terputus. • TIMEDOUT — Permintaan sesi pemain diterima, tetapi pemain tidak terhubung tidak and/or divalidasi dalam batas waktu habis (60 detik). Jenis: String Wajib: Tidak
NextToken	Token yang menunjukkan awal halaman hasil berikutnya. Untuk menentukan awal kumpulan hasil, jangan berikan nilai. Jika Anda memberikan ID sesi pemain, parameter ini diabaikan. Jenis: String Wajib: Tidak

Properti	Deskripsi
Kuota	Jumlah hasil maksimum yang akan dikembalikan. Jika Anda memberikan ID sesi pemain, parameter ini diabaikan. Jenis: <code>int</code> Wajib: Tidak

StopMatchBackfillRequest

Informasi yang digunakan untuk membatalkan permintaan pengisian ulang perjodohan. Server game mengkomunikasikan informasi ini ke Amazon GameLift Servers layanan dalam [StopMatchBackfill\(\)](#) panggilan.

Properti	Deskripsi
GameSessionArn	Pengidentifikasi sesi permainan unik dari permintaan yang dibatalkan. Jenis: <code>string</code> Wajib: Ya
MatchmakingConfigurationArn	Pengidentifikasi unik dari mak comblang permintaan ini dikirim ke. Jenis: <code>string</code> Wajib: Ya
TicketId	Pengenal unik dari tiket permintaan isi ulang yang akan dibatalkan. Jenis: <code>string</code> Wajib: Ya

GetFleetRoleCredentialsRequest

Jenis data ini memberi server game akses terbatas ke AWS sumber daya Anda yang lain. Untuk informasi selengkapnya, lihat [Menyiapkan peran layanan IAM untuk Amazon GameLift Servers](#).

Properti	Deskripsi
RoleArn	Nama Sumber Daya Amazon (ARN) dari peran layanan yang memperluas akses terbatas ke sumber daya Anda. AWS Jenis: <code>string</code> Wajib: Ya
RoleSessionName	Nama sesi yang menjelaskan penggunaan kredensial peran. Jenis: <code>string</code> Wajib: Tidak

AttributeValue

Gunakan nilai-nilai ini dalam [Pemain](#) atribut pasangan kunci-nilai. Objek ini memungkinkan Anda menentukan nilai atribut menggunakan salah satu tipe data yang valid: `string`, nomor, array `string`, atau peta data. Setiap `AttributeValue` objek hanya dapat menggunakan salah satu properti yang tersedia.

Properti	Deskripsi
ATTRType	Menentukan jenis nilai atribut. Jenis: Nilai <code>AttrType</code> enum. Wajib: Tidak
D	Merupakan nilai atribut <code>string</code>. Jenis: <code>string</code>

Properti	Deskripsi
	Wajib: Ya
T	Merupakan nilai atribut numerik. Jenis: <code>double</code> Wajib: Ya
SL	Merupakan array nilai atribut string. Jenis: <code>string[]</code> Wajib: Ya
SDM	Merupakan kamus kunci string dan nilai ganda. Jenis: <code>Dictionary<string, double></code> Wajib: Ya

AwsStringOutcome

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan. Jenis: <code>string</code> Wajib: Tidak
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: <code>bool</code> Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil.

Properti	Deskripsi
	Jenis: the section called “GameLiftError”
	Wajib: Tidak

GenericOutcome

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: bool Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Jenis: the section called “GameLiftError” Wajib: Tidak

DescribePlayerSessionsOutcome

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan. Jenis: the section called “DescribePlayerSessionsResult” Wajib: Tidak
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: bool

Properti	Deskripsi
	Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Jenis: the section called “GameLiftError” Wajib: Tidak

DescribePlayerSessionsResult

Properti	Deskripsi
NextToken	Token yang menunjukkan awal halaman hasil berikutnya. Untuk menentukan awal kumpulan hasil, jangan berikan nilai. Jika Anda memberikan ID sesi pemain, parameter ini diabaikan. Jenis: <code>string</code> Wajib: Ya
PlayerSessions	Kumpulan objek yang berisi properti untuk setiap sesi pemain yang cocok dengan permintaan. Jenis: <code>IList<the section called “PlayerSession”></code> Diperlukan:
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: <code>bool</code> Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Jenis: the section called “GameLiftError” Wajib: Tidak

PlayerSession

Properti	Deskripsi
CreationTime	Jenis: long Wajib: Ya
FleetId	Jenis: string Wajib: Ya
GameSessionId	Jenis: string Wajib: Ya
IpAddress	Jenis: string Wajib: Ya
PlayerData	Jenis: string Wajib: Ya
PlayerId	Jenis: string Wajib: Ya
PlayerSessionId	Jenis: string Wajib: Ya
Port	Jenis: int Wajib: Ya
Status	Jenis: A PlayerSessionStatus enum . Wajib: Ya
TerminationTime	Jenis: long

Properti	Deskripsi
	Wajib: Ya
DnsName	Jenis: <code>string</code> Wajib: Ya

StartMatchBackfillOutcome

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan. Jenis: the section called “StartMatchBackfillResult” Wajib: Tidak
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: <code>bool</code> Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Jenis: the section called “GameLiftError” Wajib: Tidak

StartMatchBackfillResult

Properti	Deskripsi
TicketId	Jenis: <code>string</code> Wajib: Ya

GetComputeCertificateOutcome

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan. Jenis: the section called “GetComputeCertificateResult” Wajib: Tidak
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: bool Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Jenis: the section called “GameLiftError” Wajib: Tidak

GetComputeCertificateResult

Jalur ke sertifikat TLS pada komputasi Anda dan nama host komputasi.

Properti	Deskripsi
CertificatePath	Jenis: string Wajib: Ya
ComputeName	Jenis: string Wajib: Ya

GetFleetRoleCredentialsOutcome

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan. Jenis: the section called “GetFleetRoleCredentialsResult” Wajib: Tidak
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: bool Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Jenis: the section called “GameLiftError” Wajib: Tidak

GetFleetRoleCredentialsResult

Properti	Deskripsi
AccessKeyId	ID kunci akses untuk mengautentikasi dan menyediakan akses ke AWS sumber daya Anda. Jenis: string Wajib: Tidak
AssumedRoleId	ID pengguna yang menjadi milik peran layanan. Jenis: string Wajib: Tidak

Properti	Deskripsi
AssumedRoleUserArn	Nama Sumber Daya Amazon (ARN) pengguna yang menjadi milik peran layanan. Jenis: <code>string</code> Wajib: Tidak
Kedaluwarsa	Jumlah waktu hingga kredensi sesi Anda kedaluwarsa. Jenis: <code>DateTime</code> Wajib: Tidak
SecretAccessKey	ID kunci akses rahasia untuk otentikasi. Jenis: <code>string</code> Wajib: Tidak
SessionToken	Token untuk mengidentifikasi sesi aktif saat ini yang berinteraksi dengan AWS sumber daya Anda. Jenis: <code>string</code> Wajib: Tidak
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: <code>bool</code> Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Jenis: the section called “GameLiftError” Wajib: Tidak

AwsDateTimeOutcome

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan. Jenis: <code>DateTime</code> Wajib: Tidak
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: <code>bool</code> Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Jenis: the section called “GameLiftError” Wajib: Tidak

GameLiftError

Properti	Deskripsi
ErrorType	Jenis kesalahan. Jenis: A <code>GameLiftErrorType</code> enum. Wajib: Tidak
ErrorMessage	Nama kesalahannya. Jenis: <code>string</code> Wajib: Tidak
ErrorMessage	Pesan kesalahan.

Properti	Deskripsi
	Jenis: <code>string</code>
	Wajib: Tidak

Enum

Enum yang didefinisikan untuk SDK server untuk Amazon GameLift Servers (C #) didefinisikan sebagai berikut:

AttrType

- TIDAK ADA
- TALI
- GANDA
- STRING_LIST
- STRING_DOUBLE_PETA

GameLiftErrorType

Nilai string yang menunjukkan jenis kesalahan. Nilai yang valid meliputi:

- SERVICE_CALL_FAILED - Panggilan ke layanan telah gagal. AWS
- LOCAL_CONNECTION_FAILED — Koneksi lokal gagal. Amazon GameLift Servers
- NETWORK_NOT_INITIALIZED — Jaringan belum diinisialisasi.
- GAMESESSION_ID_NOT_SET — ID sesi permainan belum ditetapkan.
- BAD_REQUEST_EXCEPTION
- INTERNAL_SERVICE_EXCEPTION
- ALREADY_INITIALIZED — Amazon GameLift Servers Server atau Klien telah diinisialisasi dengan `Initialize ()`.
- FLEET_MISMATCH — Armada target tidak cocok dengan armada `GameSession` atau `PlayerSession`.
- GAMELIFT_CLIENT_NOT_INITIALIZED — Klien belum diinisialisasi. Amazon GameLift Servers
- GAMELIFT_SERVER_NOT_INITIALIZED — Server belum diinisialisasi. Amazon GameLift Servers

- `GAME_SESSION_ENDED_FAILED` — SDK server untuk tidak Amazon GameLift Servers dapat menghubungi layanan untuk melaporkan sesi permainan berakhir.
- `GAME_SESSION_NOT_READY` — Sesi Game Server tidak diaktifkan. Amazon GameLift Servers
- `GAME_SESSION_READY_FAILED` — SDK server untuk tidak Amazon GameLift Servers dapat menghubungi layanan untuk melaporkan sesi permainan sudah siap.
- `INITIALIZATION_MISMATCH` — Metode klien dipanggil setelah `Server: :Initialize ()`, atau sebaliknya.
- `NOT_INITIALIZED` — Amazon GameLift Servers Server atau Klien belum diinisialisasi dengan `Initialize ()`.
- `NO_TARGET_ALIASID_SET` — AliasID target belum ditetapkan.
- `NO_TARGET_FLEET_SET` — Armada target belum ditetapkan.
- `PROCESS_ENDING_FAILED` — SDK server untuk tidak Amazon GameLift Servers dapat menghubungi layanan untuk melaporkan proses berakhir.
- `PROCESS_NOT_ACTIVE` — Proses server belum aktif, tidak terikat pada `GameSession`, dan tidak dapat menerima atau memproses `PlayerSessions`
- `PROCESS_NOT_READY` — Proses server belum siap untuk diaktifkan.
- `PROCESS_READY_FAILED` — SDK server untuk tidak Amazon GameLift Servers dapat menghubungi layanan untuk melaporkan proses siap.
- `SDK_VERSION_DETECTION_FAILED` — Deteksi versi SDK gagal.
- `STX_CALL_FAILED` — Panggilan ke komponen backend `XStx` server telah gagal.
- `STX_INITIALIZATION_FAILED` — Komponen backend server gagal `XStx` menginisialisasi.
- `UNEXPECTED_PLAYER_SESSION` — Sesi pemain yang tidak terdaftar ditemui oleh server.
- `WEBSOCKET_CONNECT_FAILURE`
- `WEBSOCKET_CONNECT_FAILURE_FORBIDDEN`
- `WEBSOCKET_CONNECT_FAILURE_INVALID_URL`
- `WEBSOCKET_CONNECT_FAILURE_TIMEOUT`
- `WEBSOCKET_RETRIABLE_SEND_MESSAGE_FAILURE` - Kegagalan yang dapat diambil untuk mengirim pesan ke Layanan. GameLift WebSocket
- `WEBSOCKET_SEND_MESSAGE_FAILURE` — Kegagalan untuk mengirim pesan ke Layanan. GameLift WebSocket

- MATCH_BACKFILL_REQUEST_VALIDATION — Validasi permintaan gagal.
- PLAYER_SESSION_REQUEST_VALIDATION — Validasi permintaan gagal.

PlayerSessionCreationPolicy

Nilai string yang menunjukkan apakah sesi game menerima pemain baru. Nilai yang valid meliputi:

- ACCEPT_ALL — Menerima semua sesi pemain baru.
- DENY_ALL — Menolak semua sesi pemain baru.
- NOT_SET — Sesi permainan tidak diatur untuk menerima atau menolak sesi pemain baru.

PlayerSessionStatus

- AKTIF
- SELESAI
- NOT_SET
- DIPESAN
- TIMEDOUT

C# server SDK 5.x untuk Amazon GameLift Servers - Tipe data

Topik

- [GetSdkVersion\(\)](#)
- [InitSDK\(\)](#)
- [InitSDK\(\)](#)
- [ProcessReady\(\)](#)
- [ProcessEnding\(\)](#)
- [ActivateGameSession\(\)](#)
- [UpdatePlayerSessionCreationPolicy\(\)](#)
- [GetGameSessionId\(\)](#)
- [GetTerminationTime\(\)](#)
- [AcceptPlayerSession\(\)](#)
- [RemovePlayerSession\(\)](#)

- [DescribePlayerSessions\(\)](#)
- [StartMatchBackfill\(\)](#)
- [StopMatchBackfill\(\)](#)
- [GetComputeCertificate\(\)](#)
- [GetFleetRoleCredentials\(\)](#)
- [Hancurkan \(\)](#)

GetSdkVersion()

Mengembalikan nomor versi SDK saat ini yang dibangun ke dalam proses server.

Sintaks

```
AwsStringOutcome GetSdkVersion();
```

Nilai yang ditampilkan

Jika berhasil, ini mengembalikan versi SDK saat ini sebagai objek [the section called "AwsStringOutcome"](#). String yang dikembalikan mencakup nomor versi (contoh 5.0.0). Jika tidak berhasil, ini mengembalikan pesan kesalahan.

Contoh

```
var getSdkVersionOutcome = GameLiftServerAPI.GetSdkVersion();
```

InitSDK()

Menginisialisasi Amazon GameLift Servers SDK untuk armada terkelola EC2 . Panggil metode ini saat peluncuran, sebelum inisialisasi lain yang terkait dengan Amazon GameLift Servers terjadi. Metode ini membaca parameter server dari lingkungan host untuk mengatur komunikasi antara server dan Amazon GameLift Servers layanan.

Sintaks

```
GenericOutcome InitSDK();
```

Nilai yang ditampilkan

Jika berhasil, mengembalikan `InitSdkOutcome` objek untuk menunjukkan bahwa proses server siap untuk memanggil [ProcessReady\(\)](#).

Contoh

```
//Call InitSDK to establish a local connection with the GameLift agent to enable
further communication.
GenericOutcome initSDKOutcome = GameLiftServerAPI.InitSDK();
```

InitSDK()

Menginisialisasi Amazon GameLift Servers SDK untuk armada Anywhere. Panggil metode ini saat peluncuran, sebelum inisialisasi lain yang terkait dengan Amazon GameLift Servers terjadi. Metode ini memerlukan parameter server eksplisit untuk mengatur komunikasi antara server dan Amazon GameLift Servers layanan.

Sintaks

```
GenericOutcome InitSDK(ServerParameters serverParameters);
```

Parameter

[ServerParameters](#)

Untuk menginisialisasi server game pada armada Amazon GameLift Servers Anywhere, buat `ServerParameters` objek dengan informasi berikut:

- URL yang WebSocket digunakan untuk terhubung ke server game Anda.
- ID dari proses yang digunakan untuk meng-host server game Anda.
- ID komputasi yang menghosting proses server game Anda.
- ID Amazon GameLift Servers armada yang berisi komputasi Amazon GameLift Servers Anywhere Anda.
- Token otorisasi yang dihasilkan oleh Amazon GameLift Servers operasi.

Nilai yang dikembalikan

Jika berhasil, mengembalikan `InitSdkOutcome` objek untuk menunjukkan bahwa proses server siap untuk memanggil [ProcessReady\(\)](#).

 Note

Jika panggilan gagal untuk build game yang diterapkan ke `InitSDK()` armada Anywhere, periksa `ServerSdkVersion` parameter yang digunakan saat membuat sumber daya build. Anda harus secara eksplisit menetapkan nilai ini ke versi SDK server yang digunakan. Nilai default untuk parameter ini adalah 4.x, yang tidak kompatibel. Untuk mengatasi masalah ini, buat build baru dan terapkan ke armada baru.

Contoh

```
//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()

Memberitahu Amazon GameLift Servers bahwa proses server siap untuk meng-host sesi permainan. Panggil metode ini setelah memanggil [InitSDK\(\)](#). Metode ini harus dipanggil hanya satu kali per proses.

Sintaks

```
GenericOutcome ProcessReady(ProcessParameters processParameters)
```

Parameter

[ProcessParameters](#)

Sebuah `ProcessParameters` objek menyimpan informasi tentang proses server.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Contoh ini menggambarkan baik metode dan delegasi implementasi fungsi.

```
// 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()

Memberitahu Amazon GameLift Servers bahwa proses server berakhir. Panggil metode ini setelah semua tugas pembersihan lainnya (termasuk mematikan sesi permainan aktif) dan sebelum mengakhiri proses. Tergantung pada hasil `ProcessEnding()`, proses keluar dengan sukses (0) atau kesalahan (-1) dan menghasilkan peristiwa armada. Jika proses berakhir dengan kesalahan, peristiwa armada yang dihasilkan adalah `SERVER_PROCESS_TERMINATED_UNHEALTHY`.

Sintaks

```
GenericOutcome ProcessEnding()
```

Nilai yang ditampilkan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Contoh ini memanggil `ProcessEnding()` dan `Destroy()` sebelum menghentikan proses server dengan kode keluar sukses atau kesalahan.

```
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()

Memberitahu Amazon GameLift Servers bahwa proses server telah mengaktifkan sesi permainan dan sekarang siap menerima koneksi pemain. Tindakan ini harus dipanggil sebagai bagian dari fungsi `onStartGameSession()` callback, setelah semua inisialisasi sesi game.

Sintaks

```
GenericOutcome ActivateGameSession()
```

Nilai yang ditampilkan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Contoh ini menunjukkan `ActivateGameSession()` yang dipanggil sebagai bagian dari fungsi delegasi `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()

Memperbarui kemampuan sesi game saat ini untuk menerima sesi pemain baru. Sesi game dapat diatur untuk menerima atau menolak semua sesi pemain baru.

Sintaks

```
GenericOutcome UpdatePlayerSessionCreationPolicy(PlayerSessionCreationPolicy
playerSessionPolicy)
```

Parameter

playerSessionPolicy

Nilai string yang menunjukkan apakah sesi permainan menerima pemain baru.

Nilai yang valid meliputi:

- **ACCEPT_ALL** — Menerima semua sesi pemain baru.
- **DENY_ALL** — Menolak semua sesi pemain baru.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Contoh ini menetapkan kebijakan bergabung sesi game saat ini untuk menerima semua pemain.

```
GenericOutcome updatePlayerSessionPolicyOutcome =
    GameLiftServerAPI.UpdatePlayerSessionCreationPolicy(PlayerSessionCreationPolicy.ACCEPT_ALL);
```

GetGameSessionId()

Mengambil ID sesi permainan yang dihosting oleh proses server aktif.

Untuk proses idle yang tidak diaktifkan dengan sesi game, panggilan akan menampilkan file. [the section called “GameLiftError”](#)

Sintaks

```
AwsStringOutcome GetGameSessionId()
```

Nilai yang ditampilkan

Jika berhasil, ini mengembalikan ID sesi game sebagai objek [the section called “AwsStringOutcome”](#). Jika tidak berhasil, mengembalikan pesan kesalahan.”

Contoh

```
AwsStringOutcome getGameSessionIdOutcome = GameLiftServerAPI.GetGameSessionId();
```

GetTerminationTime()

Mengembalikan waktu yang merupakan jadwal proses server akan ditutup, jika waktu penghentian tersedia. Proses server mengambil tindakan ini setelah menerima `onProcessTerminate()` panggilan balik dari Amazon GameLift Servers. Amazon GameLift Servers panggilan `onProcessTerminate()` untuk alasan berikut:

- Ketika proses server telah melaporkan kesehatan yang buruk atau belum Amazon GameLift Servers merespons.
- Saat mengakhiri instance selama acara scale-down.
- Ketika sebuah instance dihentikan karena gangguan [spot-instance](#).

Sintaks

```
AwsDateTimeOutcome GetTerminationTime()
```

Nilai yang ditampilkan

Jika berhasil, ini mengembalikan waktu penghentian sebagai objek [the section called “AwsDateTimeOutcome”](#). Nilainya adalah waktu penghentian, dinyatakan dalam kutu yang telah berlalu sejak. 0001 00:00:00 Misalnya, nilai waktu tanggal 2020-09-13 12:26:40 -000Z sama dengan 6373559680000000000 kutu. Jika tidak ada waktu penghentian tersedia, pesan kesalahan ditampilkan.

Contoh

```
AwsDateTimeOutcome getTerminationTimeOutcome = GameLiftServerAPI.GetTerminationTime();
```

AcceptPlayerSession()

Memberitahu Amazon GameLift Servers bahwa pemain dengan ID sesi pemain tertentu telah terhubung ke proses server dan membutuhkan validasi. Amazon GameLift Servers memverifikasi bahwa ID sesi pemain valid. Setelah sesi pemain divalidasi, Amazon GameLift Servers mengubah status slot pemain dari RESERVED ke ACTIVE.

Sintaks

```
GenericOutcome AcceptPlayerSession(String playerId)
```

Parameter

playerSessionId

ID unik yang dikeluarkan GameLift saat sesi pemain baru dibuat.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Contoh ini menggambarkan fungsi untuk menangani permintaan koneksi, termasuk memvalidasi dan menolak sesi pemain yang tidak valid. 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()

Memberitahu Amazon GameLift Servers bahwa pemain telah terputus dari proses server. Sebagai tanggapan, Amazon GameLift Servers ubah slot pemain menjadi tersedia.

Sintaks

```
GenericOutcome RemovePlayerSession(String playerId)
```

Parameter

playerSessionId

ID unik yang dikeluarkan Amazon GameLift Servers saat sesi pemain baru dibuat.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

```
GenericOutcome removePlayerSessionOutcome =  
    GameLiftServerAPI.RemovePlayerSession(playerSessionId);
```

DescribePlayerSessions()

Mengambil data sesi pemain yang mencakup pengaturan, metadata sesi, dan data pemain. Gunakan tindakan ini untuk mendapatkan informasi untuk satu sesi pemain, untuk semua sesi pemain dalam sesi game, atau untuk semua sesi pemain yang terkait dengan ID pemain tunggal.

Sintaks

```
DescribePlayerSessionsOutcome DescribePlayerSessions(DescribePlayerSessionsRequest  
    describePlayerSessionsRequest)
```

Parameter

[DescribePlayerSessionsRequest](#)

[the section called “DescribePlayerSessionsRequest”](#) Objek yang menggambarkan sesi pemain mana yang akan diambil.

Nilai yang dikembalikan

Jika berhasil, mengembalikan [the section called “DescribePlayerSessionsOutcome”](#) objek yang berisi satu set objek sesi pemain yang sesuai dengan parameter permintaan.

Contoh

Contoh ini menggambarkan permintaan untuk semua sesi pemain yang secara aktif terhubung ke sesi game tertentu. Dengan menghilangkan NextToken dan mengatur nilai Batas ke 10, Amazon GameLift Servers akan mengembalikan catatan sesi 10 pemain pertama yang cocok dengan permintaan.

```
// 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()

Mengirim permintaan untuk menemukan pemain baru untuk slot terbuka dalam sesi permainan yang dibuat dengan FlexMatch. Untuk informasi selengkapnya, lihat [fitur FlexMatch isi ulang](#).

Tindakan ini asinkron. Jika pemain baru dicocokkan, kirimkan data Amazon GameLift Servers mak comblang yang diperbarui menggunakan fungsi panggilan balik. `OnUpdateGameSession()`

Proses server hanya dapat melakukan satu permintaan backfill match yang aktif dalam satu waktu. Untuk mengirim permintaan baru, panggil [StopMatchBackfill\(\)](#) terlebih dahulu untuk membatalkan permintaan asli.

Sintaks

```
StartMatchBackfillOutcome StartMatchBackfill (StartMatchBackfillRequest  
startBackfillRequest);
```

Parameter

[StartMatchBackfillRequest](#)

Sebuah `StartMatchBackfillRequest` objek menyimpan informasi tentang permintaan isi ulang.

Nilai yang dikembalikan

Mengembalikan [the section called “StartMatchBackfillOutcome”](#) objek dengan ID tiket isi ulang pertandingan, atau kegagalan dengan pesan kesalahan.

Contoh

```
// 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()

Membatalkan permintaan pengisian ulang pertandingan yang aktif. Untuk informasi selengkapnya, lihat [fitur FlexMatch isi ulang](#).

Sintaks

```
GenericOutcome StopMatchBackfill (StopMatchBackfillRequest stopBackfillRequest);
```

Parameter

[StopMatchBackfillRequest](#)

StopMatchBackfillRequestObjek yang memberikan detail tentang tiket perijodohan yang Anda hentikan.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

```
// 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()

Mengambil jalur ke sertifikat TLS yang digunakan untuk mengenkripsi koneksi jaringan antara server game dan klien game Anda. Anda dapat menggunakan jalur sertifikat saat mendaftarkan perangkat komputasi ke armada Amazon GameLift Servers Anywhere. Untuk informasi selengkapnya, lihat [RegisterCompute](#).

Sintaks

```
GetComputeCertificateOutcome GetComputeCertificate();
```

Nilai yang ditampilkan

Mengembalikan `GetComputeCertificateResponse` objek yang berisi berikut:

- `CertificatePath`: Jalur ke sertifikat TLS pada sumber daya komputasi Anda. Saat menggunakan armada Amazon GameLift Servers terkelola, jalur ini berisi:
 - `certificate.pem`: Sertifikat pengguna akhir. Rantai sertifikat lengkap adalah kombinasi dari `certificateChain.pem` ditambahkan ke sertifikat ini.
 - `certificateChain.pem`: Rantai sertifikat yang berisi sertifikat root dan sertifikat perantara.
 - `rootCertificate.pem`: Sertifikat root.
 - `privateKey.pem`: Kunci pribadi untuk sertifikat pengguna akhir.
- `ComputeName`: Nama sumber daya komputasi Anda.

Contoh

```
GetComputeCertificateOutcome getComputeCertificateOutcome =  
    GameLiftServerAPI.GetComputeCertificate();
```

GetFleetRoleCredentials()

Mengambil kredensial peran IAM yang mengizinkan Amazon GameLift Servers untuk berinteraksi dengan orang lain. Layanan AWS Untuk informasi selengkapnya, lihat [Berkomunikasi dengan AWS sumber daya lain dari armada Anda](#).

Sintaks

```
GetFleetRoleCredentialsOutcome GetFleetRoleCredentials(GetFleetRoleCredentialsRequest request);
```

Parameter

[GetFleetRoleCredentialsRequest](#)

Kredensial peran yang memperluas akses terbatas ke AWS sumber daya Anda ke server game.

Nilai yang dikembalikan

Mengembalikan objek [the section called “GetFleetRoleCredentialsOutcome”](#).

Contoh

```
// form the fleet credentials request
GetFleetRoleCredentialsRequest getFleetRoleCredentialsRequest = new
    GetFleetRoleCredentialsRequest(){
        RoleArn = "arn:aws:iam::123456789012:role/service-role/exampleGameLiftAction"
    };
GetFleetRoleCredentialsOutcome GetFleetRoleCredentialsOutcome credentials =
    GetFleetRoleCredentials(getFleetRoleCredentialsRequest);
```

Hancurkan ()

Membebaskan SDK server Amazon GameLift Servers game dari memori. Sebagai praktik terbaik, hubungi metode ini setelah `ProcessEnding()` dan sebelum mengakhiri proses. Jika Anda menggunakan armada Anywhere dan Anda tidak menghentikan proses server setelah setiap sesi game, panggil `Destroy()` dan kemudian `InitSDK()` untuk menginisialisasi ulang sebelum memberi tahu Amazon GameLift Servers bahwa proses tersebut siap untuk menjadi tuan rumah sesi game. `ProcessReady()`

Sintaks

```
GenericOutcome Destroy()
```

Nilai yang ditampilkan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

```
// Operations to end game sessions and the server process
GenericOutcome processEndingOutcome = GameLiftServerAPI.ProcessEnding();

// Shut down and destroy the instance of the GameLift Game Server SDK
GenericOutcome destroyOutcome = GameLiftServerAPI.Destroy();

// Exit the process with success or failure
if (processEndingOutcome.Success)
{
    Environment.Exit(0);
}
else
{
    Console.WriteLine("ProcessEnding() failed. Error: " +
        processEndingOutcome.Error.ToString());
    Environment.Exit(-1);
}
```

Go server SDK untuk Amazon GameLift Servers -- Tindakan

Gunakan referensi SDK 5.x server untuk mengintegrasikan game multipemain Anda untuk hosting. Amazon GameLift Servers Untuk panduan tentang proses integrasi, lihat [Tambahkan Amazon GameLift Servers ke server game Anda](#).

`GameLiftServerAPI.gomendefinisikan` tindakan SDK server Go.

Go server SDK untuk Amazon GameLift Servers - Tipe data

Gunakan referensi SDK server untuk mengintegrasikan game multipemain Anda untuk hosting. Amazon GameLift Servers Untuk panduan tentang proses integrasi, lihat [Tambahkan Amazon GameLift Servers ke server game Anda](#).

`GameLiftServerAPI.gomendefinisikan` tindakan SDK server Go.

[Go server SDK untuk Amazon GameLift Servers -- Tindakan](#)

Jenis Data

- [LogParameters](#)
- [ProcessParameters](#)
- [UpdateGameSession](#)
- [GameSession](#)
- [ServerParameters](#)
- [StartMatchBackfillRequest](#)
- [Pemain](#)
- [DescribePlayerSessionsRequest](#)
- [StopMatchBackfillRequest](#)
- [GetFleetRoleCredentialsRequest](#)

LogParameters

Objek yang mengidentifikasi file yang dihasilkan selama sesi permainan yang Amazon GameLift Servers ingin Anda unggah dan simpan setelah sesi permainan berakhir. Server game menyediakan LogParameters Amazon GameLift Servers sebagai bagian dari ProcessParameters objek dalam [ProcessReady\(\)](#) panggilan.

Sifat-sifat	Deskripsi
LogPaths	Daftar path direktori ke file log server game yang ingin Anda simpan Amazon GameLift Servers untuk akses future. Proses server menghasilkan file-file ini selama setiap sesi permainan. Anda menentukan jalur dan nama file di server game Anda dan menyimpannya di direktori build game root. Jalur log harus absolut. Misalnya, jika build game Anda menyimpan log sesi game di jalur seperti <code>MyGame\sessionLogs\</code> , maka jalurnya akan berada <code>c:\game\MyGame\sessionLogs</code> di instance Windows. Jenis: []string Wajib: Tidak

ProcessParameters

Objek yang menggambarkan komunikasi antara proses server dan Amazon GameLift Servers. Proses server menyediakan informasi ini Amazon GameLift Servers dengan panggilan ke [ProcessReady\(\)](#).

Sifat-sifat	Deskripsi
LogParameters	Objek dengan jalur direktori ke file yang dihasilkan selama sesi permainan. Amazon GameLift Servers menyalin dan menyimpan file untuk akses future. Jenis: LogParameters Wajib: Tidak
OnHealthCheck	Fungsi callback yang Amazon GameLift Servers memanggil untuk meminta laporan status kesehatan dari proses server. Amazon GameLift Servers memanggil fungsi ini setiap 60 detik dan menunggu 60 detik untuk respons. Proses server kembali TRUE jika sehat, FALSE jika tidak sehat. Jika tidak ada respons yang dikembalikan, Amazon GameLift Servers mencatat proses server sebagai tidak sehat. Jenis: OnHealthCheck func() bool Wajib: Tidak
OnProcessTerminate	Fungsi callback yang Amazon GameLift Servers memanggil untuk memaksa proses server dimatikan. Setelah memanggil fungsi ini, Amazon GameLift Servers tunggu 5 menit hingga proses server dimatikan dan merespons dengan ProcessEnding() panggilan sebelum mematikan proses server. Jenis: OnProcessTerminate func() Wajib: Ya
OnStartGameSession	Fungsi callback yang Amazon GameLift Servers memanggil untuk meneruskan objek sesi game yang diperbarui ke proses server. Amazon GameLift Servers memanggil fungsi ini ketika permintaan pengisian ulang kecocokan telah diproses untuk menyediakan data mak comblang yang diperbarui. Ini melewati GameSession objek, pembaruan status (updateReason), dan ID tiket isi ulang pertandingan.

	Jenis: <code>OnStartGameSession func (model.GameSession)</code> Wajib: Ya
OnUpdateGameSession	Fungsi callback yang Amazon GameLift Servers memanggil untuk meneruskan informasi sesi game yang diperbarui ke proses server. Amazon GameLift Servers memanggil fungsi ini setelah memproses permintaan pengisian ulang kecocokan untuk menyediakan data mak comblang yang diperbarui. Jenis: <code>OnUpdateGameSession func (model.UpdateGameSession)</code> Wajib: Tidak
Port	Nomor port yang didengarkan oleh proses server untuk koneksi pemain baru. Nilai harus masuk ke dalam kisaran port yang dikonfigurasi untuk setiap armada yang men-deploy build server game ini. Nomor port ini termasuk dalam sesi game dan objek sesi pemain, yang digunakan sesi game saat menghubungkan ke proses server. Jenis: <code>int</code> Wajib: Ya

UpdateGameSession

Pembaruan untuk objek sesi permainan, yang mencakup alasan bahwa sesi permainan diperbarui, dan ID tiket isi ulang terkait jika pengisian ulang digunakan untuk mengisi sesi pemain dalam sesi permainan.

Properti	Deskripsi
GameSession	Sebuah objek GameSession. GameSession Objek berisi properti yang menggambarkan sesi permainan. Jenis: <code>GameSession GameSession()</code> Wajib: Ya
UpdateReason	Alasan bahwa sesi permainan sedang diperbarui.

Properti	Deskripsi
	Jenis: UpdateReason UpdateReason() Wajib: Ya
BackfillTicketId	ID tiket isi ulang yang mencoba memperbarui sesi permainan. Jenis: String Wajib: Tidak

GameSession

Detail sesi permainan.

Properti	Deskripsi
GameSessionId	Pengenal unik untuk sesi permainan. Sesi permainan Nama Sumber Daya Amazon (ARN) memiliki format berikut: <code>arn:aws:gamelift:<region>::gamesession/<fleet ID>/<custom ID string or idempotency token></code> Jenis: String Wajib: Tidak
Nama	Label deskriptif dari sesi permainan. Jenis: String Wajib: Tidak
FleetId	Pengenal unik untuk armada tempat sesi permainan berjalan. Jenis: String Wajib: Tidak

Properti	Deskripsi
MaximumPlayerSessionCount	Jumlah maksimum koneksi pemain ke sesi permainan. Jenis: Integer Wajib: Tidak
Port	Nomor port untuk sesi permainan. Untuk terhubung ke server Amazon GameLift Servers game, aplikasi memerlukan alamat IP dan nomor port. Jenis: Integer Wajib: Tidak
IpAddress	Alamat IP dari sesi game. Untuk terhubung ke server Amazon GameLift Servers game, aplikasi memerlukan alamat IP dan nomor port. Jenis: String Wajib: Tidak
GameSessionData	Set properti sesi game khusus, diformat sebagai nilai string tunggal. Jenis: String Wajib: Tidak
MatchmakerData	Informasi tentang proses perjodohan yang digunakan untuk membuat sesi permainan, dalam sintaks JSON, diformat sebagai string. Selain konfigurasi perjodohan yang digunakan, ini berisi data tentang semua pemain yang ditugaskan untuk pertandingan, termasuk atribut pemain dan tugas tim. Jenis: String Wajib: Tidak

Properti	Deskripsi
GameProperties	Satu set properti kustom untuk sesi permainan, diformat sebagai pasangan key:value. Properti ini diteruskan dengan permintaan untuk memulai sesi permainan baru. Jenis: <code>map[string] string</code> Wajib: Tidak
DnsName	Pengenal DNS yang ditetapkan ke instance yang menjalankan sesi permainan. Nilai memiliki format berikut: - Armada yang mendukung TLS:<unique identifier>.<region identifier>.amazongamelift.com . - Non-TLS-enabled armada:ec2-<unique identifier>.compute.amazonaws.com . Saat menghubungkan ke sesi permainan yang berjalan pada armada yang mendukung TLS, Anda harus menggunakan nama DNS, bukan alamat IP. Jenis: <code>String</code> Wajib: Tidak

ServerParameters

Informasi yang digunakan untuk menjaga koneksi antara server Amazon GameLift Servers Anywhere dan Amazon GameLift Servers layanan. Informasi ini digunakan saat meluncurkan proses server baru dengan [InitSDK\(\)](#). Untuk server yang dihosting pada EC2 instance Amazon GameLift Servers terkelola, gunakan objek kosong.

Properti	Deskripsi
WebSocket URL	<code>GameLiftServerSdkEndpoint</code> Amazon GameLift ServersPengembalian ketika Anda RegisterCompute untuk sumber daya komputasi Amazon GameLift Servers Anywhere.

Properti	Deskripsi
	Jenis: <code>string</code> Wajib: Ya
ProcessID	Pengenal unik yang terdaftar pada proses server yang menghosting game Anda. Jenis: <code>string</code> Wajib: Ya
HostID	Pengidentifikasi unik dari sumber daya komputasi yang menghosting proses server baru. HostID ini adalah yang <code>ComputeName</code> digunakan saat Anda mendaftarkan komputasi Anda. Untuk informasi selengkapnya, lihat RegisterCompute . Jenis: <code>string</code> Wajib: Ya
FleetID	Pengidentifikasi unik armada tempat komputasi terdaftar. Untuk informasi selengkapnya, lihat RegisterCompute . Jenis: <code>string</code> Wajib: Ya
AuthToken	Token otentikasi yang dihasilkan oleh Amazon GameLift Servers itu mengotentikasi server Anda. Amazon GameLift Servers Untuk informasi selengkapnya, lihat GetComputeAuthToken . Jenis: <code>string</code> Wajib: Ya

StartMatchBackfillRequest

Informasi yang digunakan untuk membuat permintaan pengisian ulang perjodohan. Server game mengkomunikasikan informasi ini Amazon GameLift Servers dalam [StartMatchBackfill\(\)](#) panggilan.

Properti	Deskripsi
GameSessionArn	Pengidentifikasi sesi permainan yang unik. Operasi API GetGameSessionId mengembalikan pengenal dalam format ARN. Jenis: <code>String</code> Wajib: Ya
MatchmakingConfigurationArn	Pengidentifikasi unik (dalam bentuk ARN) untuk mak comblang untuk digunakan untuk permintaan ini. ARN mak comblang untuk sesi permainan asli ada di objek sesi permainan di properti data mak comblang. Untuk informasi selengkapnya tentang data mak comblang, lihat Bekerja dengan data mak comblang. Jenis: <code>String</code> Wajib: Ya
Pemain	Satu set data yang mewakili semua pemain yang saat ini dalam sesi permainan. Matchmaker menggunakan informasi ini untuk mencari pemain baru yang cocok untuk pemain saat ini. Jenis: <code>[]model.Player</code> Wajib: Ya
TicketId	Pengenal unik untuk tiket permintaan pencocokan atau pencocokan isi ulang. Jika Anda tidak memberikan nilai, Amazon GameLift Servers hasilkan satu. Gunakan pengidentifikasi ini untuk melacak status tiket backfill match atau membatalkan permintaan jika diperlukan. Jenis: <code>String</code> Wajib: Tidak

Pemain

Objek yang mewakili pemain dalam perjudohan. Ketika permintaan perjudohan dimulai, pemain memiliki ID pemain, atribut, dan mungkin data latensi. Amazon GameLift Servers menambahkan informasi tim setelah pertandingan dibuat.

Properti	Deskripsi
LatencyInMS	Satu set nilai yang dinyatakan dalam milidetik yang menunjukkan jumlah latensi yang dialami pemain saat terhubung ke suatu lokasi. Jika properti ini digunakan, pemain hanya cocok untuk lokasi yang terdaftar. Jika mak comblang memiliki aturan yang mengevaluasi latensi pemain, pemain harus melaporkan latensi untuk dicocokkan. Jenis: <code>map[string] int</code> Wajib: Tidak
PlayerAttributes	Kumpulan pasangan kunci:value yang berisi informasi pemain untuk digunakan dalam perjudohan. Kunci atribut pemain harus cocok dengan yang PlayerAttributes digunakan dalam set aturan perjudohan. Untuk informasi selengkapnya tentang atribut pemain, lihat AttributeValue. Jenis: <code>map[string] AttributeValue</code> Wajib: Tidak
PlayerId	Pengenal unik untuk pemain. Jenis: <code>String</code> Wajib: Tidak
Tim	Nama tim yang ditugaskan pemain dalam pertandingan. Anda menentukan nama tim dalam set aturan perjudohan. Jenis: <code>String</code> Wajib: Tidak

DescribePlayerSessionsRequest

Objek yang menentukan sesi pemain mana yang akan diambil. Proses server menyediakan informasi ini dengan [DescribePlayerSessions\(\)](#) panggilan keAmazon GameLift Servers.

Properti	Deskripsi
GameSessionID	Pengidentifikasi sesi permainan yang unik. Gunakan parameter ini untuk meminta semua sesi pemain untuk sesi game yang ditentukan. Format ID sesi game adalah <code>arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string></code> . <code>GameSessionID</code> Ini adalah string ID kustom atau string yang dihasilkan. Jenis: <code>String</code> Wajib: Tidak
PlayerSessionID	Pengenal unik untuk sesi pemain. Gunakan parameter ini untuk meminta satu sesi pemain tertentu. Jenis: <code>String</code> Wajib: Tidak
PlayerID	Pengenal unik untuk pemain. Gunakan parameter ini untuk meminta semua sesi pemain untuk pemain tertentu. Lihat Hasilkan pemain IDs. Jenis: <code>String</code> Wajib: Tidak
PlayerSessionStatusFilter	Status sesi pemain untuk memfilter hasil. Status sesi pemain yang mungkin meliputi: • <code>RESERVED</code> — Permintaan sesi pemain diterima, tetapi pemain belum terhubung ke proses server atau telah divalidasi. • <code>AKTIF</code> — Pemain divalidasi oleh proses server dan terhubung. • <code>SELESAI</code> — Koneksi pemain terputus.

Properti	Deskripsi
	TIMEDOUT — Permintaan sesi pemain diterima, tetapi pemain tidak terhubung atau tidak divalidasi dalam batas waktu habis (60 detik). Jenis: <code>String</code> Wajib: Tidak
<code>NextToken</code>	Token yang menunjukkan awal halaman hasil berikutnya. Untuk menentukan awal kumpulan hasil, jangan berikan nilai. Jika Anda memberikan ID sesi pemain, parameter ini diabaikan. Jenis: <code>String</code> Wajib: Tidak
<code>Limit</code>	Jumlah hasil maksimum yang akan dikembalikan. Jika Anda memberikan ID sesi pemain, parameter ini diabaikan. Jenis: <code>int</code> Wajib: Tidak

StopMatchBackfillRequest

Informasi yang digunakan untuk membatalkan permintaan pengisian ulang perjodohan. Server game mengkomunikasikan informasi ini ke Amazon GameLift Servers layanan dalam [StopMatchBackfill\(\)](#) panggilan.

Properti	Deskripsi
<code>GameSessionArn</code>	Pengidentifikasi sesi permainan unik dari permintaan yang dibatalkan. Jenis: <code>string</code> Wajib: Tidak

Properti	Deskripsi
MatchmakingConfigurationArn	Pengidentifikasi unik dari mak comblang permintaan ini dikirim ke. Jenis: <code>string</code> Wajib: Tidak
TicketId	Pengenal unik dari tiket permintaan isi ulang yang akan dibatalkan. Jenis: <code>string</code> Wajib: Tidak

GetFleetRoleCredentialsRequest

Kredensi peran yang memperluas akses terbatas ke AWS sumber daya Anda ke server game. Untuk informasi selengkapnya, lihat [Menyiapkan peran layanan IAM untuk Amazon GameLift Servers](#).

Properti	Deskripsi
RoleArn	ARN dari peran layanan yang memperluas akses terbatas ke sumber daya Anda. AWS Jenis: <code>string</code> Wajib: Ya
RoleSessionName	Nama sesi yang menjelaskan penggunaan kredensi peran. Jenis: <code>string</code> Wajib: Ya

[Go server SDK untuk Amazon GameLift Servers - Tipe data](#)

Tindakan

- [GetSdkVersion\(\)](#)
- [InitSDK\(\)](#)

- [ProcessReady\(\)](#)
- [ProcessEnding\(\)](#)
- [ActivateGameSession\(\)](#)
- [UpdatePlayerSessionCreationPolicy\(\)](#)
- [GetGameSessionId\(\)](#)
- [GetTerminationTime\(\)](#)
- [AcceptPlayerSession\(\)](#)
- [RemovePlayerSession\(\)](#)
- [DescribePlayerSessions\(\)](#)
- [StartMatchBackfill\(\)](#)
- [StopMatchBackfill\(\)](#)
- [GetComputeCertificate\(\)](#)
- [GetFleetRoleCredentials\(\)](#)
- [Hancurkan \(\)](#)

GetSdkVersion()

Mengembalikan nomor versi SDK saat ini yang dibangun ke dalam proses server.

Sintaks

```
func GetSdkVersion() (string, error)
```

Nilai yang ditampilkan

Jika berhasil, mengembalikan versi SDK saat ini sebagai string. String yang dikembalikan mencakup nomor versi (contoh `5.0.0`). Jika tidak berhasil, mengembalikan pesan kesalahan seperti `common.SdkVersionDetectionFailed`.

Contoh

```
version, err := server.GetSdkVersion()
```

InitSDK()

Menginisialisasi Amazon GameLift Servers SDK. Panggil metode ini saat peluncuran sebelum inisialisasi lain yang terkait dengan Amazon GameLift Servers terjadi. Metode ini mengatur komunikasi antara server dan Amazon GameLift Servers layanan.

Sintaks

```
func InitSDK(params ServerParameters) error
```

Parameter

ServerParameters

Untuk menginisialisasi server game pada armada Amazon GameLift Servers Anywhere, buat `ServerParameters` objek dengan informasi berikut:

- URL yang WebSocket digunakan untuk terhubung ke server game Anda.
- ID dari proses yang digunakan untuk meng-host server game Anda.
- ID komputasi yang menghosting proses server game Anda.
- ID Amazon GameLift Servers armada yang berisi komputasi Amazon GameLift Servers Anywhere Anda.
- Token otorisasi yang dihasilkan oleh Amazon GameLift Servers operasi.

Untuk menginisialisasi server game pada EC2 armada yang Amazon GameLift Servers dikelola, buat `ServerParameters` objek tanpa parameter. Dengan panggilan ini, Amazon GameLift Servers agen mengatur lingkungan komputasi dan secara otomatis terhubung ke Amazon GameLift Servers layanan untuk Anda.

Nilai yang dikembalikan

Jika berhasil, mengembalikan `nil` kesalahan untuk menunjukkan bahwa proses server siap untuk memanggil [ProcessReady\(\)](#).

Note

Jika panggilan gagal untuk build game yang diterapkan ke `InitSDK()` armada Anywhere, periksa `ServerSdkVersion` parameter yang digunakan saat membuat sumber daya build. Anda harus secara eksplisit menetapkan nilai ini ke versi SDK server yang digunakan. Nilai

default untuk parameter ini adalah 4.x, yang tidak kompatibel. Untuk mengatasi masalah ini, buat build baru dan terapkan ke armada baru.

Contoh

Amazon GameLift Servers Contoh di mana saja

```
//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 contoh terkelola

```
//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()

Memberitahu Amazon GameLift Servers bahwa proses server siap untuk meng-host sesi permainan. Panggil metode ini setelah memanggil [InitSDK\(\)](#). Metode ini harus dipanggil hanya satu kali per proses.

Sintaks

```
func ProcessReady(param ProcessParameters) error
```

Parameter

ProcessParameters

[ProcessParameters](#) Objek mengkomunikasikan informasi berikut tentang proses server:

- Nama-nama metode callback yang diimplementasikan dalam kode server game yang dipanggil Amazon GameLift Servers layanan untuk berkomunikasi dengan proses server.
- Nomor port yang didengarkan oleh proses server.
- Tipe [LogParameters](#) data yang berisi jalur ke file khusus sesi game apa pun yang Amazon GameLift Servers ingin Anda ambil dan simpan.

Nilai yang dikembalikan

Mengembalikan kesalahan dengan pesan kesalahan jika metode gagal. Mengembalikan `nil` jika metode ini berhasil.

Contoh

Contoh ini menggambarkan panggilan [ProcessReady\(\)](#) dan mendelegasi implementasi fungsi.

```
// 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()

Memberitahu Amazon GameLift Servers bahwa proses server berakhir. Panggil metode ini setelah semua tugas pembersihan lainnya (termasuk mematikan sesi permainan aktif) dan sebelum mengakhiri proses. Tergantung pada hasil `ProcessEnding()`, proses keluar dengan sukses (0)

atau kesalahan (-1) dan menghasilkan peristiwa armada. Jika proses berakhir dengan kesalahan, peristiwa armada yang dihasilkan adalah `SERVER_PROCESS_TERMINATED_UNHEALTHY`.

Sintaks

```
func ProcessEnding() error
```

Nilai yang ditampilkan

Mengembalikan kode kesalahan 0 atau kode kesalahan yang ditentukan.

Contoh

```
// 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()

Memberitahu Amazon GameLift Servers bahwa proses server telah mengaktifkan sesi permainan dan sekarang siap menerima koneksi pemain. Tindakan ini disebut sebagai bagian dari fungsi `onStartGameSession()` callback, setelah semua inisialisasi sesi game.

Sintaks

```
func ActivateGameSession() error
```

Nilai yang ditampilkan

Mengembalikan kesalahan dengan pesan kesalahan jika metode gagal.

Contoh

Contoh ini menunjukkan `ActivateGameSession()` dipanggil sebagai bagian dari fungsi `onStartGameSession()` delegasi.

```
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()

Memperbarui kemampuan sesi game saat ini untuk menerima sesi pemain baru. Sesi game dapat diatur untuk menerima atau menolak semua sesi pemain baru.

Sintaks

```
func UpdatePlayerSessionCreationPolicy(policy model.PlayerSessionCreationPolicy) error
```

Parameter

playerSessionCreationKebijakan

Nilai string yang menunjukkan apakah sesi permainan menerima pemain baru.

Nilai yang valid meliputi:

- **model.AcceptAll**— Terima semua sesi pemain baru.
- **model.DenyAll**— Tolak semua sesi pemain baru.

Nilai yang dikembalikan

Mengembalikan kesalahan dengan pesan kesalahan jika terjadi kegagalan.

Contoh

Contoh ini menetapkan kebijakan bergabung sesi game saat ini untuk menerima semua pemain.

```
err := server.UpdatePlayerSessionCreationPolicy(model.AcceptAll)
```

GetGameSessionId()

Mengambil ID sesi permainan yang dihosting oleh proses server aktif.

Sintaks

```
func GetGameSessionID() (string, error)
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai yang dikembalikan

Jika berhasil, mengembalikan ID sesi permainan dan kesalahan nihil. Untuk proses idle yang belum diaktifkan dengan sesi permainan, panggilan mengembalikan string kosong dan `nil` kesalahan.

Contoh

```
gameSessionID, err := server.GetGameSessionID()
```

GetTerminationTime()

Mengembalikan waktu proses server dijadwalkan untuk dimatikan jika waktu penghentian tersedia. Proses server mengambil tindakan ini setelah menerima `onProcessTerminate()` panggilan balik dari Amazon GameLift Servers. Amazon GameLift Servers panggilan `onProcessTerminate()` untuk alasan berikut:

- Ketika proses server telah melaporkan kesehatan yang buruk atau belum Amazon GameLift Servers merespons.
- Saat mengakhiri instance selama acara scale-down.
- Ketika sebuah instance dihentikan karena gangguan [spot-instance](#).

Sintaks

```
func GetTerminationTime() (int64, error)
```

Nilai yang ditampilkan

Jika berhasil, mengembalikan stempel waktu dalam detik epoch yang proses server dijadwalkan untuk dimatikan dan penghentian kesalahan. `nil` Nilainya adalah waktu penghentian, dinyatakan dalam kutu yang telah berlalu dari. `0001 00:00:00` Misalnya, nilai waktu tanggal `2020-09-13 12:26:40 -000Z` sama dengan `637355968000000000` kutu. Jika tidak ada waktu penghentian tersedia, pesan kesalahan ditampilkan.

Contoh

```
terminationTime, err := server.GetTerminationTime()
```

AcceptPlayerSession()

Memberitahu Amazon GameLift Servers bahwa pemain dengan ID sesi pemain tertentu telah terhubung ke proses server dan membutuhkan validasi. Amazon GameLift Servers memverifikasi bahwa ID sesi pemain valid. Setelah sesi pemain divalidasi, Amazon GameLift Servers ubah status slot pemain dari `RESERVED` ke `ACTIVE`.

Sintaks

```
func AcceptPlayerSession(playerSessionID string) error
```

Parameter

playerSessionId

ID unik yang dikeluarkan Amazon GameLift Servers saat sesi pemain baru dibuat.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Contoh ini menangani permintaan koneksi yang mencakup memvalidasi dan menolak sesi pemain yang tidak valid. IDs

```
func ReceiveConnectingPlayerSessionID(conn Connection, playerSessionID string) {
```

```
err := server.AcceptPlayerSession(playerSessionID)
if err != nil {
    connection.Accept()
} else {
    connection.Reject(err.Error())
}
}
```

RemovePlayerSession()

Memberitahu Amazon GameLift Servers bahwa pemain telah terputus dari proses server. Sebagai tanggapan, Amazon GameLift Servers ubah slot pemain menjadi tersedia.

Sintaks

```
func RemovePlayerSession(playerSessionID string) error
```

Parameter

playerSessionId

ID unik yang dikeluarkan Amazon GameLift Servers saat sesi pemain baru dibuat.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

```
err := server.RemovePlayerSession(playerSessionID)
```

DescribePlayerSessions()

Mengambil data sesi pemain yang mencakup pengaturan, metadata sesi, dan data pemain. Gunakan metode ini untuk mendapatkan informasi tentang hal-hal berikut:

- Sesi pemain tunggal
- Semua sesi pemain dalam sesi permainan
- Semua sesi pemain yang terkait dengan ID pemain tunggal

Sintaks

```
func DescribePlayerSessions(req request.DescribePlayerSessionsRequest)
    (result.DescribePlayerSessionsResult, error) {
    return srv.describePlayerSessions(&req)
}
```

Parameter

[DescribePlayerSessionsRequest](#)

Sebuah `DescribePlayerSessionsRequest` objek menjelaskan sesi pemain mana yang akan diambil.

Nilai yang dikembalikan

Jika berhasil, mengembalikan `DescribePlayerSessionsResult` objek yang berisi satu set objek sesi pemain yang sesuai dengan parameter permintaan.

Contoh

Contoh ini meminta semua sesi pemain yang terhubung secara aktif ke sesi permainan tertentu. Dengan menghilangkan `NextToken` dan mengatur nilai `Batas` ke 10, Amazon GameLift Servers mengembalikan catatan sesi 10 pemain pertama yang cocok dengan permintaan.

```
// 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()

Mengirim permintaan untuk menemukan pemain baru untuk slot terbuka dalam sesi permainan yang dibuat dengan `FlexMatch`. Untuk informasi selengkapnya, lihat [fitur FlexMatch isi ulang](#).

Tindakan ini asinkron. Jika pemain baru dicocokkan, kirimkan data Amazon GameLift Servers mak comblang yang diperbarui menggunakan fungsi panggilan balik. `OnUpdateGameSession()`

Proses server hanya dapat melakukan satu permintaan backfill match yang aktif dalam satu waktu. Untuk mengirim permintaan baru, panggil [StopMatchBackfill\(\)](#) terlebih dahulu untuk membatalkan permintaan asli.

Sintaks

```
func StartMatchBackfill(req request.StartMatchBackfillRequest)
    (result.StartMatchBackfillResult, error)
```

Parameter

[StartMatchBackfillRequest](#)

Sebuah `StartMatchBackfillRequest` objek mengkomunikasikan informasi berikut:

- ID tiket untuk ditetapkan ke permintaan backfill. Informasi ini opsional; jika tidak ada ID yang diberikan, Amazon GameLift Servers hasilkan satu.
- Matchmaker untuk dikirim permintaan. ARN konfigurasi penuh diperlukan. Nilai ini ada dalam data mak comblang sesi permainan.
- ID sesi permainan untuk mengisi ulang.
- Data perjodohan yang tersedia untuk pemain sesi permainan saat ini.

Nilai yang dikembalikan

Mengembalikan `StartMatchBackfillResult` objek dengan ID tiket isi ulang kecocokan, atau kegagalan dengan pesan kesalahan.

Contoh

```
// 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()

Membatalkan permintaan pengisian ulang pertandingan yang aktif. Untuk informasi selengkapnya, lihat [fitur FlexMatch isi ulang](#).

Sintaks

```
func StopMatchBackfill(req request.StopMatchBackfillRequest) error
```

Parameter

[StopMatchBackfillRequest](#)

StopMatchBackfillRequest Objek yang mengidentifikasi tiket perjodohan untuk dibatalkan:

- ID tiket yang ditetapkan untuk permintaan pengisian ulang.
- Mak comblang permintaan isi ulang dikirim ke.
- Sesi permainan terkait dengan permintaan isi ulang.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

```
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()

Mengambil jalur ke sertifikat TLS yang digunakan untuk mengenkripsi koneksi jaringan antara server game dan klien game Anda. Anda dapat menggunakan jalur sertifikat saat mendaftarkan perangkat komputasi ke armada Amazon GameLift Servers Anywhere. Untuk informasi selengkapnya, lihat [RegisterCompute](#).

Sintaks

```
func GetComputeCertificate() (result.GetComputeCertificateResult, error)
```

Nilai yang ditampilkan

Mengembalikan `GetComputeCertificateResult` objek yang berisi berikut:

- `CertificatePath`: Jalur ke sertifikat TLS pada sumber daya komputasi Anda. Saat menggunakan armada Amazon GameLift Servers terkelola, jalur ini berisi:
 - `certificate.pem`: Sertifikat pengguna akhir. Rantai sertifikat lengkap adalah kombinasi dari `certificateChain.pem` ditambahkan ke sertifikat ini.
 - `certificateChain.pem`: Rantai sertifikat yang berisi sertifikat root dan sertifikat perantara.
 - `rootCertificate.pem`: Sertifikat root.
 - `privateKey.pem`: Kunci pribadi untuk sertifikat pengguna akhir.
- `ComputeName`: Nama sumber daya komputasi Anda.

Contoh

```
tlsCertificate, err := server.GetFleetRoleCredentials(getFleetRoleCredentialsRequest)
```

GetFleetRoleCredentials()

Mengambil kredensial peran layanan yang Anda buat untuk memperluas izin ke orang lain. Layanan AWS Amazon GameLift Servers Kredensial ini memungkinkan server game Anda menggunakan

sumber daya Anda AWS . Untuk informasi selengkapnya, lihat [Menyiapkan peran layanan IAM untuk Amazon GameLift Servers](#).

Sintaks

```
func GetFleetRoleCredentials(  
    req request.GetFleetRoleCredentialsRequest,  
) (result.GetFleetRoleCredentialsResult, error) {  
    return srv.getFleetRoleCredentials(&req)  
}
```

Parameter

[GetFleetRoleCredentialsRequest](#)

Kredensial peran yang memperluas akses terbatas ke AWS sumber daya Anda ke server game.

Nilai yang dikembalikan

Mengembalikan `GetFleetRoleCredentialsResult` objek yang berisi berikut:

- `AssumedRoleUserArn` - Nama Sumber Daya Amazon (ARN) pengguna yang menjadi milik peran layanan.
- `AssumedRoleId` - ID pengguna yang menjadi milik peran layanan.
- `AccessKeyId` - ID kunci akses untuk mengautentikasi dan menyediakan akses ke AWS sumber daya Anda.
- `SecretAccessKey` - ID kunci akses rahasia untuk otentikasi.
- `SessionToken` - Token untuk mengidentifikasi sesi aktif saat ini yang berinteraksi dengan AWS sumber daya Anda.
- `Kedaluwarsa` - Jumlah waktu hingga kredensial sesi Anda kedaluwarsa.

Contoh

```
// form the customer credentials request  
getFleetRoleCredentialsRequest := request.NewGetFleetRoleCredentials()  
getFleetRoleCredentialsRequest.RoleArn = "arn:aws:iam::123456789012:role/service-role/  
exampleGameLiftAction"
```

```
credentials, err := server.GetFleetRoleCredentials(getFleetRoleCredentialsRequest)
```

Hancurkan ()

Membebaskan SDK server Amazon GameLift Servers game dari memori. Sebagai praktik terbaik, hubungi metode ini setelah `ProcessEnding()` dan sebelum mengakhiri proses. Jika Anda menggunakan armada Anywhere dan Anda tidak menghentikan proses server setelah setiap sesi game, panggil `Destroy()` dan kemudian `InitSDK()` untuk menginisialisasi ulang sebelum memberi tahu Amazon GameLift Servers bahwa proses tersebut siap untuk menjadi tuan rumah sesi game. `ProcessReady()`

Sintaks

```
func Destroy() error {  
    return srv.destroy()  
}
```

Nilai yang ditampilkan

Mengembalikan kesalahan dengan pesan kesalahan jika metode gagal.

Contoh

```
// operations to end game sessions and the server process  
defer func() {  
    err := server.ProcessEnding()  
    server.Destroy()  
    if err != nil {  
        fmt.Println("ProcessEnding() failed. Error: ", err)  
        os.Exit(-1)  
    } else {  
        os.Exit(0)  
    }  
}
```

Server C ++ (Tidak nyata) SDK 5.x untuk -- Tindakan Amazon GameLift Servers

Gunakan referensi Amazon GameLift Servers Unreal server SDK 5.x untuk membantu Anda mempersiapkan game multipemain untuk digunakan. Amazon GameLift Servers Untuk detail

tentang proses integrasi, lihat [Tambahkan Amazon GameLift Servers ke server game Anda](#). Jika Anda menggunakan Amazon GameLift Servers plugin untuk Unreal, lihat juga [Amazon GameLift Servers plugin untuk Unreal Engine](#).

 Note

Topik ini menjelaskan Amazon GameLift Servers C++ API yang dapat Anda gunakan saat membuat Unreal Engine. Secara khusus, dokumentasi ini berlaku untuk kode yang Anda kompilasi dengan `-DBUILD_FOR_UNREAL=1` opsi.

Server C ++ (Tidak nyata) SDK 5.x untuk Amazon GameLift Servers -- Tipe data

Gunakan referensi Amazon GameLift Servers Unreal server SDK 5.x untuk membantu Anda mempersiapkan game multipemain untuk digunakan. Amazon GameLift Servers Untuk detail tentang proses integrasi, lihat [Tambahkan Amazon GameLift Servers ke server game Anda](#). Jika Anda menggunakan Amazon GameLift Servers plugin untuk Unreal, lihat juga [Amazon GameLift Servers plugin untuk Unreal Engine](#).

 Note

Topik ini menjelaskan Amazon GameLift Servers C++ API yang dapat Anda gunakan saat membuat Unreal Engine. Secara khusus, dokumentasi ini berlaku untuk kode yang Anda kompilasi dengan `-DBUILD_FOR_UNREAL=1` opsi.

[Server C ++ \(Tidak nyata\) SDK 5.x untuk -- Tindakan Amazon GameLift Servers](#)

Jenis Data

- [FProcessParameter](#)
- [UpdateGameSession](#)
- [GameSession](#)
- [FServerParameter](#)
- [FStartMatchBackfillRequest](#)
- [FPlayer](#)
- [FGameLiftDescribePlayerSessionsRequest](#)
- [FStopMatchBackfillRequest](#)

- [FAttributeNilai](#)
- [FGameLiftGetFleetRoleCredentialsRequest](#)
- [FGameLiftLongOutcome](#)
- [FGameLiftStringOutcome](#)
- [FGameLiftDescribePlayerSessionsOutcome](#)
- [FGameLiftDescribePlayerSessionsResult](#)
- [FGenericHasil](#)
- [FGameLiftPlayerSession](#)
- [FGameLiftGetComputeCertificateOutcome](#)
- [FGameLiftGetComputeCertificateResult](#)
- [FGameLiftGetFleetRoleCredentialsOutcome](#)
- [FGetFleetRoleCredentialsResult](#)
- [FGameLiftError](#)
- [Enum](#)

FProcessParameter

Tipe data ini berisi kumpulan parameter yang dikirim ke Amazon GameLift Servers dalam file [ProcessReady\(\)](#).

Sifat-sifat	Deskripsi
LogParameters	Objek dengan jalur direktori ke file yang dihasilkan selama sesi permainan. Amazon GameLift Servers menyalin dan menyimpan file untuk akses future. Jenis: TArray<FString> Wajib: Tidak
OnHealthCheck	Fungsi callback yang Amazon GameLift Servers memanggil untuk meminta laporan status kesehatan dari proses server. Amazon GameLift Servers memanggil fungsi ini setiap

60 detik dan menunggu 60 detik untuk respons. Proses server kembali TRUE jika sehat, FALSE jika tidak sehat. Jika tidak ada respons yang dikembalikan, Amazon GameLift Servers mencatat proses server sebagai tidak sehat.

Properti ini adalah fungsi delegasi yang didefinisikan sebagai `DECLARE_DELEGATE_RetVal(bool, FOnHealthCheck)` ;

Jenis: `FOnHealthCheck`

Wajib: Tidak

OnProcessTerminate

Fungsi callback yang Amazon GameLift Servers memanggil untuk memaksa proses server dimatikan. Setelah memanggil fungsi ini, Amazon GameLift Servers tunggu 5 menit hingga proses server dimatikan dan merespons dengan [ProcessEnding\(\)](#) panggilan sebelum mematikan proses server.

Jenis: `FSimpleDelegate`

Wajib: Ya

OnStartGameSession

Fungsi callback yang Amazon GameLift Servers memanggil untuk mengaktifkan sesi permainan baru. Amazon GameLift Servers memanggil fungsi ini sebagai tanggapan atas permintaan klien [CreateGameSession](#). Fungsi callback melewati [GameSession](#) objek.

Properti ini adalah fungsi delegasi didefinisikan sebagai `DECLARE_DELEGATE_OneParam(FOnStartGameSession, Aws::GameLift::Server::Model::GameSession);`

Jenis: `FOnStartGameSession`

Wajib: Ya

OnUpdateGameSession

Fungsi callback yang Amazon GameLift Servers memanggil untuk meneruskan objek sesi game yang diperbarui ke proses server. Amazon GameLift Servers memanggil fungsi ini ketika permintaan pengisian ulang kecocokan telah diproses untuk menyediakan data mak comblang yang diperbarui. Ini melewati [GameSession](#) objek, pembaruan status (`updateReason`), dan ID tiket isi ulang pertandingan.

Properti ini adalah fungsi delegasi didefinisikan sebagai `DECLARE_DELEGATE_OneParam(FOnUpdateGameSession, Aws::GameLift::Server::Model::UpdateGameSession);`

Jenis: `FOnUpdateGameSession`

Wajib: Tidak

Port

Nomor port yang didengarkan oleh proses server untuk koneksi pemain baru. Nilai harus masuk ke dalam kisaran port yang dikonfigurasi untuk setiap armada yang men-deploy build server game ini. Nomor port ini termasuk dalam sesi game dan objek sesi pemain, yang digunakan sesi game saat menghubungkan ke proses server.

Jenis: `int`

Wajib: Ya

UpdateGameSession

Jenis data ini diperbarui ke objek sesi permainan, yang mencakup alasan bahwa sesi permainan diperbarui dan ID tiket isi ulang terkait jika pengisian ulang digunakan untuk mengisi sesi pemain dalam sesi permainan.

Properti	Deskripsi
GameSession	Sebuah objek GameSession. <code>GameSession</code> Objek berisi properti yang menggambarkan sesi permainan. Jenis: <code>Aws::GameLift::Server::GameSession</code> Wajib: Tidak
UpdateReason	Alasan bahwa sesi permainan sedang diperbarui. Jenis: <code>enum class UpdateReason</code> • <code>MATCHMAKING_DATA_DIPERBARUI</code> • <code>BACKFILL_FAILED</code> • <code>BACKFILL_TIMED_OUT</code>

Properti	Deskripsi
	BACKFILL_CANCELLED Wajib: Tidak
BackfillTicketId	ID tiket isi ulang yang mencoba memperbarui sesi permainan. Jenis: <code>char[]</code> Wajib: Tidak

GameSession

Tipe data ini memberikan detail sesi permainan.

Properti	Deskripsi
GameSessionId	Pengenal unik untuk sesi permainan. Sesi permainan ARN memiliki format berikut: <code>arn:aws:gamelift:<region>::gamesession/<fleet ID>/<custom ID string or idempotency token></code> Jenis: <code>char[]</code> Wajib: Tidak
Nama	Label deskriptif dari sesi permainan. Jenis: <code>char[]</code> Wajib: Tidak
FleetId	Pengenal unik untuk armada tempat sesi permainan berjalan. Jenis: <code>char[]</code>

Properti	Deskripsi
	Wajib: Tidak
MaximumPlayerSessionCount	Jumlah maksimum koneksi pemain ke sesi permainan. Jenis: <code>int</code> Wajib: Tidak
Port	Nomor port untuk sesi permainan. Untuk terhubung ke server Amazon GameLift Servers game, aplikasi memerlukan alamat IP dan nomor port. Jenis: <code>int</code> Wajib: Tidak
IpAddress	Alamat IP dari sesi game. Untuk terhubung ke server Amazon GameLift Servers game, aplikasi memerlukan alamat IP dan nomor port. Jenis: <code>char[]</code> Wajib: Tidak
GameSessionData	Set properti sesi game khusus, diformat sebagai nilai string tunggal. Jenis: <code>char[]</code> Wajib: Tidak

Properti	Deskripsi
MatchmakerData	Informasi tentang proses perjodohan yang digunakan untuk membuat sesi permainan, dalam sintaks JSON, diformat sebagai string. Selain konfigurasi perjodohan yang digunakan , ini berisi data tentang semua pemain yang ditugaskan untuk pertandingan, termasuk atribut pemain dan tugas tim. Jenis: <code>char[]</code> Wajib: Tidak
GameProperties	Satu set properti kustom untuk sesi permainan, diformat sebagai pasangan key:value. Properti ini diteruskan dengan permintaan untuk memulai sesi permainan baru. Jenis: <code>GameProperty[]</code> Wajib: Tidak

Properti	Deskripsi
DnsName	Pengenalan DNS yang ditetapkan ke instance yang menjalankan sesi permainan. Nilai memiliki format berikut: - Armada yang mendukung TLS:<unique identifier>.<region identifier>.amazongamelift.com . - Non-TLS-enabled armada:ec2-<unique identifier>.compute.amazonaws.com . Saat menghubungkan ke sesi permainan yang berjalan pada armada yang mendukung TLS, Anda harus menggunakan nama DNS, bukan alamat IP. Jenis: char[] Wajib: Tidak

FServerParameter

Informasi yang digunakan untuk menjaga koneksi antara server Amazon GameLift Servers Anywhere dan Amazon GameLift Servers layanan. Informasi ini digunakan saat meluncurkan proses server baru dengan [InitSDK\(\)](#). Untuk server yang dihosting pada EC2 instance Amazon GameLift Servers terkelola, gunakan objek kosong.

Properti	Deskripsi
webSocketUrl	GameLiftServerSdkEndpoint Amazon GameLift ServersPengembalian ketika Anda RegisterCompute untuk sumber daya komputasi Amazon GameLift Servers Anywhere.

Properti	Deskripsi
	Jenis: <code>char[]</code> Wajib: Ya
ProsesSid	Pengenal unik yang terdaftar pada proses server yang menghosting game Anda. Jenis: <code>char[]</code> Wajib: Ya
hostId	HostIDIni adalah yang <code>ComputeName</code> digunakan saat Anda mendaftarkan komputasi Anda. Untuk informasi selengkapnya, lihat RegisterCompute. Jenis: <code>char[]</code> Wajib: Ya
FleeTID	Pengidentifikasi unik armada tempat komputasi terdaftar. Untuk informasi selengkapnya, lihat RegisterCompute. Jenis: <code>char[]</code> Wajib: Ya
AuthToken	Token otentikasi yang dihasilkan oleh Amazon GameLift Servers itu mengotentikasi server Anda ke. Amazon GameLift Servers Untuk informasi selengkapnya, lihat GetComputeAuthToken. Jenis: <code>char[]</code> Wajib: Ya

FStartMatchBackfillRequest

Informasi yang digunakan untuk membuat permintaan pengisian ulang perjodohan. Server game mengkomunikasikan informasi ini Amazon GameLift Servers dalam [StartMatchBackfill\(\)](#) panggilan.

Properti	Deskripsi
GameSessionArn	Pengidentifikasi sesi permainan yang unik. Operasi API GetGameSessionId mengembalikan pengenal dalam format ARN. Jenis: <code>char[]</code> Wajib: Ya
MatchmakingConfigurationArn	Pengidentifikasi unik, dalam bentuk ARN, untuk digunakan mak comblang untuk permintaan ini. ARN mak comblang untuk sesi permainan asli ada di objek sesi permainan di properti data mak comblang. Pelajari selengkapnya tentang data matchmaker di Bekerja dengan data matchmaker. Jenis: <code>char[]</code> Wajib: Ya
Pemain	Satu set data yang mewakili semua pemain yang berada di sesi permainan. Matchmaker menggunakan informasi ini untuk mencari pemain baru yang cocok untuk pemain saat ini. Jenis: <code>TArray<FPlayer></code> Wajib: Ya
TicketId	Pengenal unik untuk tiket permintaan pencocokan atau pencocokan isi ulang. Jika Anda tidak memberikan nilai, Amazon GameLift Servers hasilkan satu. Gunakan pengident

Properti	Deskripsi
	ifikasi ini untuk melacak status tiket backfill match atau membatalkan permintaan jika diperlukan. Jenis: <code>char[]</code> Wajib: Tidak

FPlayer

Tipe data ini mewakili pemain dalam perjodohan. Saat memulai permintaan perjodohan, pemain memiliki ID pemain, atribut, dan mungkin data latensi. Amazon GameLift Servers menambahkan informasi tim setelah pertandingan dibuat.

Properti	Deskripsi
LatencyInMS	Satu set nilai yang dinyatakan dalam milidetik yang menunjukkan jumlah latensi yang dialami pemain saat terhubung ke suatu lokasi. Jika properti ini digunakan, pemain hanya cocok untuk lokasi yang terdaftar. Jika mak comblang memiliki aturan yang mengevaluasi latensi pemain, pemain harus melaporkan latensi untuk dicocokkan. Jenis: <code>TMap>FString, int32<</code> Wajib: Tidak
PlayerAttributes	Kumpulan pasangan kunci:nilai yang berisi informasi pemain untuk digunakan dalam perjodohan. Kunci atribut pemain harus cocok dengan yang PlayerAttributes digunakan dalam set aturan perjodohan.

Properti	Deskripsi
	Untuk informasi selengkapnya tentang atribut pemain, lihat AttributeValue. Jenis: TMap>FString, FAttributeValue< Wajib: Tidak
PlayerId	Pengenal unik untuk pemain. Jenis: std::string Wajib: Tidak
Tim	Nama tim yang ditugaskan pemain dalam pertandingan. Anda menentukan nama tim dalam set aturan perjodohan. Jenis: FString Wajib: Tidak

FGameLiftDescribePlayerSessionsRequest

Objek yang menentukan sesi pemain mana yang akan diambil. Proses server menyediakan informasi ini dengan [DescribePlayerSessions\(\)](#) panggilan keAmazon GameLift Servers.

Properti	Deskripsi
GameSessionId	Pengidentifikasi sesi permainan yang unik. Gunakan parameter ini untuk meminta semua sesi pemain untuk sesi game yang ditentukan. Format ID sesi game adalahFString. GameSessionID Ini adalah string ID kustom atau Jenis: std::string

Properti	Deskripsi
	Wajib: Tidak
PlayerSessionId	Pengenal unik untuk sesi pemain. Gunakan parameter ini untuk meminta satu sesi pemain tertentu. Jenis: FString Wajib: Tidak
PlayerId	Pengenal unik untuk pemain. Gunakan parameter ini untuk meminta semua sesi pemain untuk pemain tertentu. Lihat Hasilkan pemain IDs. Jenis: FString Wajib: Tidak
PlayerSessionStatusFilter	Status sesi pemain untuk memfilter hasil. Status sesi pemain yang mungkin meliputi: • RESERVED — Permintaan sesi pemain diterima, tetapi pemain belum terhubung ke proses server atau telah divalidasi. • AKTIF — Pemain divalidasi oleh proses server dan terhubung. • SELESAI — Koneksi pemain terputus. • TIMEDOUT — Permintaan sesi pemain diterima, tetapi pemain tidak terhubung atau tidak divalidasi dalam batas waktu habis (60 detik). Jenis: FString Wajib: Tidak

Properti	Deskripsi
NextToken	Token yang menunjukkan awal halaman hasil berikutnya. Untuk menentukan awal kumpulan hasil, jangan berikan nilai. Jika Anda memberikan ID sesi pemain, parameter ini diabaikan. Jenis: FString Wajib: Tidak
Kuota	Jumlah hasil maksimum yang akan dikembalikan. Jika Anda memberikan ID sesi pemain, parameter ini diabaikan. Jenis: int Wajib: Tidak

FStopMatchBackfillRequest

Informasi yang digunakan untuk membatalkan permintaan pengisian ulang perjodohan. Server game mengkomunikasikan informasi ini ke Amazon GameLift Servers layanan dalam [StopMatchBackfill\(\)](#) panggilan.

Properti	Deskripsi
GameSessionArn	Pengidentifikasi sesi permainan unik dari permintaan yang dibatalkan. Jenis: FString Wajib: Ya
MatchmakingConfigurationArn	Pengidentifikasi unik dari mak comblang permintaan ini dikirim ke. Jenis: FString

Properti	Deskripsi
	Wajib: Ya
TicketId	Pengenal unik dari tiket permintaan isi ulang yang akan dibatalkan. Jenis: FString Wajib: Ya

FAttributeNilai

Gunakan nilai-nilai ini dalam [FPlayer](#) atribut pasangan kunci-nilai. Objek ini memungkinkan Anda menentukan nilai atribut menggunakan salah satu tipe data yang valid: string, nomor, array string, atau peta data. Setiap `AttributeValue` objek hanya dapat menggunakan salah satu properti yang tersedia.

Properti	Deskripsi
ATTRType	Menentukan jenis nilai atribut. Jenis: Nilai <code>FAttributeType</code> enum. Wajib: Tidak
D	Merupakan nilai atribut string. Jenis: FString Wajib: Tidak
T	Merupakan nilai atribut numerik. Jenis: double Wajib: Tidak
SL	Merupakan array nilai atribut string. Jenis: TArray<FString>

Properti	Deskripsi
	Wajib: Tidak
SDM	Merupakan kamus kunci string dan nilai ganda. Jenis: TMap<FString, double> Wajib: Tidak

FGameLiftGetFleetRoleCredentialsRequest

Tipe data ini menyediakan kredensi peran yang memperluas akses terbatas ke AWS sumber daya Anda ke server game. Untuk informasi selengkapnya, lihat [Menyiapkan peran layanan IAM untuk Amazon GameLift Servers](#).

Properti	Deskripsi
RoleArn	Nama Sumber Daya Amazon (ARN) dari peran layanan yang memperluas akses terbatas ke sumber daya Anda. AWS Jenis: FString Wajib: Tidak
RoleSessionName	Nama sesi yang menjelaskan penggunaan kredensi peran. Jenis: FString Wajib: Tidak

FGameLiftLongOutcome

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan. Jenis: long Wajib: Tidak
ResultWithOwnership	Hasil dari tindakan, dilemparkan sebagai rvalue, sehingga kode panggilan dapat mengambil kepemilikan objek. Jenis: long&& Wajib: Tidak
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: bool Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Jenis: the section called “FGameLiftError” Wajib: Tidak

FGameLiftStringOutcome

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan. Jenis: FString Wajib: Tidak
ResultWithOwnership	Hasil dari tindakan, dilemparkan sebagai rvalue, sehingga kode panggilan dapat mengambil kepemilikan objek.

Properti	Deskripsi
	Jenis: FString&& Wajib: Tidak
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: bool Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Jenis: the section called “FGameLiftError” Wajib: Tidak

FGameLiftDescribePlayerSessionsOutcome

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan. Jenis: the section called “FGameLiftDescribePlayerSessionsResult” Wajib: Tidak
ResultWithOwnership	Hasil dari tindakan, dilemparkan sebagai rvalue, sehingga kode panggilan dapat mengambil kepemilikan objek. Jenis: FGameLiftDescribePlayerSessionsResult&& Wajib: Tidak
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: bool

Properti	Deskripsi
	Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Jenis: the section called “FGameLiftError” Wajib: Tidak

FGameLiftDescribePlayerSessionsResult

Properti	Deskripsi
PlayerSessions	Jenis: TArray<FGameLiftPlayerSession> Wajib: Ya
NextToken	Token yang menunjukkan awal halaman hasil berikutnya. Untuk menentukan awal kumpulan hasil, jangan berikan nilai. Jika Anda memberikan ID sesi pemain, parameter ini diabaikan. Jenis: FString Wajib: Tidak
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: bool Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Jenis: the section called “FGameLiftError” Wajib: Tidak

FGenericHasil

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: bool Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Jenis: the section called “FGameLiftError” Wajib: Tidak

FGameLiftPlayerSession

Properti	Deskripsi
CreationTime	Jenis: long Wajib: Ya
FleetId	Jenis: FString Wajib: Ya
GameSessionId	Jenis: FString Wajib: Ya
IpAddress	Jenis: FString Wajib: Ya
PlayerData	Jenis: FString Wajib: Ya

Properti	Deskripsi
PlayerId	Jenis: FString Wajib: Ya
PlayerSessionId	Jenis: FString Wajib: Ya
Port	Jenis: int Wajib: Ya
Status	Tipe: A PlayerSessionStatus enum . Wajib: Ya
TerminationTime	Jenis: long Wajib: Ya
DnsName	Jenis: FString Wajib: Ya

FGameLiftGetComputeCertificateOutcome

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan. Jenis: the section called “FGameLiftGetComputeCertificateResult” Wajib: Tidak
ResultWithOwnership	Hasil dari tindakan, dilemparkan sebagai rvalue, sehingga kode panggilan dapat mengambil kepemilikan objek.

Properti	Deskripsi
	Jenis: FGameLiftGetComputeCertificateResult&& Wajib: Tidak
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: bool Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Jenis: the section called “FGameLiftError” Wajib: Tidak

FGameLiftGetComputeCertificateResult

Jalur ke sertifikat TLS pada komputasi Anda dan nama host komputasi.

Properti	Deskripsi
CertificatePath	Jenis: FString Wajib: Ya
ComputeName	Jenis: FString Wajib: Ya

FGameLiftGetFleetRoleCredentialsOutcome

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan.

Properti	Deskripsi
	Jenis: the section called “FGetFleetRoleCredentialsResult” Wajib: Tidak
ResultWithOwnership	Hasil dari tindakan, dilemparkan sebagai rvalue, sehingga kode panggilan dapat mengambil kepemilikan objek. Jenis: FGameLiftGetFleetRoleCredentialsResult&& Wajib: Tidak
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: bool Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Jenis: the section called “FGameLiftError” Wajib: Tidak

FGetFleetRoleCredentialsResult

Properti	Deskripsi
AccessKeyId	ID kunci akses untuk mengautentikasi dan menyediakan akses ke AWS sumber daya Anda. Jenis: FString Wajib: Tidak
AssumedRoleId	ID pengguna yang menjadi milik peran layanan. Jenis: FString

Properti	Deskripsi
	Wajib: Tidak
AssumedRoleUserArn	Nama Sumber Daya Amazon (ARN) pengguna yang menjadi milik peran layanan. Jenis: FString Wajib: Tidak
Kedaluwarsa	Jumlah waktu hingga kredensi sesi Anda kedaluwarsa. Jenis: FDateTime Wajib: Tidak
SecretAccessKey	ID kunci akses rahasia untuk otentikasi. Jenis: FString Wajib: Tidak
SessionToken	Token untuk mengidentifikasi sesi aktif saat ini yang berinteraksi dengan AWS sumber daya Anda. Jenis: FString Wajib: Tidak
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: bool Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Jenis: the section called “GameLiftError” Wajib: Tidak

FGameLiftError

Properti	Deskripsi
ErrorType	Jenis kesalahan. Tipe: A GameLiftErrorType enum. Wajib: Tidak
ErrorMessage	Pesan kesalahan. Jenis: std::string Wajib: Tidak

Enum

Enum yang didefinisikan untuk SDK server untuk Amazon GameLift Servers (Unreal) didefinisikan sebagai berikut:

FAttributeTipe

- TIDAK ADA
- STRING
- GANDA
- STRING_LIST
- STRING_DOUBLE_PETA

GameLiftErrorType

Nilai string yang menunjukkan jenis kesalahan. Nilai yang valid meliputi:

- SERVICE_CALL_FAILED - Panggilan ke layanan telah gagal. AWS
- LOCAL_CONNECTION_FAILED — Koneksi lokal gagal. Amazon GameLift Servers

- NETWORK_NOT_INITIALIZED — Jaringan belum diinisialisasi.
- GAMESESSION_ID_NOT_SET — ID sesi permainan belum ditetapkan.
- BAD_REQUEST_EXCEPTION
- INTERNAL_SERVICE_EXCEPTION
- ALREADY_INITIALIZED — Amazon GameLift Servers Server atau Klien telah diinisialisasi dengan Initialize ().
- FLEET_MISMATCH — Armada target tidak cocok dengan armada GameSession atau PlayerSession.
- GAMELIFT_CLIENT_NOT_INITIALIZED — Klien belum diinisialisasi. Amazon GameLift Servers
- GAMELIFT_SERVER_NOT_INITIALIZED — Server belum diinisialisasi. Amazon GameLift Servers
- GAME_SESSION_ENDED_FAILED — SDK server untuk tidak Amazon GameLift Servers dapat menghubungi layanan untuk melaporkan sesi permainan berakhir.
- GAME_SESSION_NOT_READY — Sesi Game Server tidak diaktifkan. Amazon GameLift Servers
- GAME_SESSION_READY_FAILED — SDK server untuk tidak Amazon GameLift Servers dapat menghubungi layanan untuk melaporkan sesi permainan sudah siap.
- INITIALIZATION_MISMATCH — Metode klien dipanggil setelah Server: :Initialize (), atau sebaliknya.
- NOT_INITIALIZED — Amazon GameLift Servers Server atau Klien belum diinisialisasi dengan Initialize ().
- NO_TARGET_ALIASID_SET — AliaID target belum ditetapkan.
- NO_TARGET_FLEET_SET — Armada target belum ditetapkan.
- PROCESS_ENDING_FAILED — SDK server untuk tidak Amazon GameLift Servers dapat menghubungi layanan untuk melaporkan proses berakhir.
- PROCESS_NOT_ACTIVE — Proses server belum aktif, tidak terikat pada GameSession, dan tidak dapat menerima atau memproses. PlayerSessions
- PROCESS_NOT_READY — Proses server belum siap untuk diaktifkan.
- PROCESS_READY_FAILED — SDK server untuk tidak Amazon GameLift Servers dapat menghubungi layanan untuk melaporkan proses siap.
- SDK_VERSION_DETECTION_FAILED — Deteksi versi SDK gagal.
- STX_CALL_FAILED — Panggilan ke komponen backend XStx server telah gagal.

- STX_INITIALIZATION_FAILED — Komponen backend server gagal XStx menginisialisasi.
- UNEXPECTED_PLAYER_SESSION — Sesi pemain yang tidak terdaftar ditemui oleh server.
- WEBSOCKET_CONNECT_FAILURE
- WEBSOCKET_CONNECT_FAILURE_FORBIDDEN
- WEBSOCKET_CONNECT_FAILURE_INVALID_URL
- WEBSOCKET_CONNECT_FAILURE_TIMEOUT
- WEBSOCKET_RETRIABLE_SEND_MESSAGE_FAILURE - Kegagalan yang dapat diambil untuk mengirim pesan ke Layanan. GameLift WebSocket
- WEBSOCKET_SEND_MESSAGE_FAILURE — Kegagalan untuk mengirim pesan ke Layanan. GameLift WebSocket
- MATCH_BACKFILL_REQUEST_VALIDATION — Validasi permintaan gagal.
- PLAYER_SESSION_REQUEST_VALIDATION — Validasi permintaan gagal.

EPlayerSessionCreationPolicy

Nilai string yang menunjukkan apakah sesi game menerima pemain baru. Nilai yang valid meliputi:

- ACCEPT_ALL — Menerima semua sesi pemain baru.
- DENY_ALL — Menolak semua sesi pemain baru.
- NOT_SET — Sesi permainan tidak diatur untuk menerima atau menolak sesi pemain baru.

EPlayerSessionStatus

- AKTIF
- SELESAI
- NOT_SET
- DIPESAN
- TIMEDOUT

[Server C ++ \(Tidak nyata\) SDK 5.x untuk Amazon GameLift Servers -- Tipe data](#)

Topik

- [GetSdkVersion\(\)](#)
- [InitSDK\(\)](#)

- [InitSDK\(\)](#)
- [ProcessReady\(\)](#)
- [ProcessEnding\(\)](#)
- [ActivateGameSession\(\)](#)
- [UpdatePlayerSessionCreationPolicy\(\)](#)
- [GetGameSessionId\(\)](#)
- [GetTerminationTime\(\)](#)
- [AcceptPlayerSession\(\)](#)
- [RemovePlayerSession\(\)](#)
- [DescribePlayerSessions\(\)](#)
- [StartMatchBackfill\(\)](#)
- [StopMatchBackfill\(\)](#)
- [GetComputeCertificate\(\)](#)
- [GetFleetRoleCredentials\(\)](#)

GetSdkVersion()

Mengembalikan nomor versi SDK saat ini yang dibangun ke dalam proses server.

Sintaks

```
FGameLiftStringOutcome GetSdkVersion();
```

Nilai yang ditampilkan

Jika berhasil, ini mengembalikan versi SDK saat ini sebagai objek [the section called "FGameLiftStringOutcome"](#). Objek yang dikembalikan menyertakan nomor versi (contoh 5.0.0). Jika tidak berhasil, ini mengembalikan pesan kesalahan.

Contoh

```
Aws::GameLift::AwsStringOutcome SdkVersionOutcome =  
    Aws::GameLift::Server::GetSdkVersion();
```

InitSDK()

Menginisialisasi Amazon GameLift Servers SDK untuk armada terkelola EC2 . Panggil metode ini saat peluncuran, sebelum inisialisasi lain yang terkait dengan Amazon GameLift Servers terjadi. Metode ini membaca parameter server dari lingkungan host untuk mengatur komunikasi antara server dan Amazon GameLift Servers layanan.

Sintaks

```
FGameLiftGenericOutcome InitSDK()
```

Nilai yang ditampilkan

Jika berhasil, mengembalikan InitSdkOutcome objek yang menunjukkan bahwa proses server siap untuk memanggil [ProcessReady\(\)](#).

Contoh

```
//Call InitSDK to establish a local connection with the Amazon GameLift Servers Agent  
to enable further communication.  
FGameLiftGenericOutcome initSdkOutcome = gameLiftSdkModule->InitSDK();
```

InitSDK()

Menginisialisasi Amazon GameLift Servers SDK untuk armada Anywhere atau armada kontainer terkelola. Panggil metode ini saat peluncuran, sebelum inisialisasi lain yang terkait dengan Amazon GameLift Servers terjadi. Metode ini memerlukan parameter server eksplisit untuk mengatur komunikasi antara server dan Amazon GameLift Servers layanan.

Sintaks

```
FGameLiftGenericOutcome InitSDK(serverParameters)
```

Parameter

[FServerParameter](#)

Untuk menginisialisasi server game pada armada Amazon GameLift Servers Anywhere, buat `ServerParameters` objek dengan informasi berikut:

- URL yang WebSocket digunakan untuk terhubung ke server game Anda.
- ID dari proses yang digunakan untuk meng-host server game Anda.
- ID komputasi yang menghosting proses server game Anda.
- ID Amazon GameLift Servers armada yang berisi komputasi Amazon GameLift Servers Anywhere Anda.
- Token otorisasi yang dihasilkan oleh Amazon GameLift Servers operasi.

Nilai yang dikembalikan

Jika berhasil, mengembalikan `InitSdkOutcome` objek yang menunjukkan bahwa proses server siap untuk memanggil [ProcessReady\(\)](#).

Note

Jika panggilan gagal untuk build game yang diterapkan ke `InitSDK()` armada Anywhere, periksa `ServerSdkVersion` parameter yang digunakan saat membuat sumber daya build. Anda harus secara eksplisit menetapkan nilai ini ke versi SDK server yang digunakan. Nilai default untuk parameter ini adalah 4.x, yang tidak kompatibel. Untuk mengatasi masalah ini, buat build baru dan terapkan ke armada baru.

Contoh

```
//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()

Memberitahu Amazon GameLift Servers bahwa proses server siap untuk meng-host sesi permainan. Panggil metode ini setelah memanggil [InitSDK\(\)](#). Metode ini harus dipanggil hanya satu kali per proses.

Sintaks

```
GenericOutcome ProcessReady(const Aws::GameLift::Server::ProcessParameters
&processParameters);
```

Parameter

processParameters

[FProcessParameter](#) Objek yang mengkomunikasikan informasi berikut tentang proses server:

- Nama metode callback yang diimplementasikan dalam kode server game yang dipanggil Amazon GameLift Servers layanan untuk berkomunikasi dengan proses server.
- Nomor port yang didengarkan oleh proses server.
- Jalur ke file khusus sesi game apa pun yang Amazon GameLift Servers ingin Anda tangkap dan simpan.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Contoh ini menggambarkan panggilan [ProcessReady\(\)](#) dan mendelegasi implementasi fungsi.

```
//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()

Memberitahu Amazon GameLift Servers bahwa proses server berakhir. Panggil metode ini setelah semua tugas pembersihan lainnya (termasuk mematikan sesi permainan aktif) dan sebelum

mengakhiri proses. Tergantung pada hasil `ProcessEnding()`, proses keluar dengan sukses (0) atau kesalahan (-1) dan menghasilkan peristiwa armada. Jika proses berakhir dengan kesalahan, peristiwa armada yang dihasilkan adalah `SERVER_PROCESS_TERMINATED_UNHEALTHY`).

Sintaks

```
FGameLiftGenericOutcome ProcessEnding()
```

Nilai yang ditampilkan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

```
//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()

Memberitahu Amazon GameLift Servers bahwa proses server telah mengaktifkan sesi permainan dan sekarang siap menerima koneksi pemain. Tindakan ini harus dipanggil sebagai bagian dari fungsi `onStartGameSession()` callback, setelah semua inisialisasi sesi game.

Sintaks

```
FGameLiftGenericOutcome ActivateGameSession()
```

Nilai yang ditampilkan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Contoh ini menunjukkan `ActivateGameSession()` dipanggil sebagai bagian dari fungsi `onStartGameSession()` delegasi.

```
//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()

Memperbarui kemampuan sesi game saat ini untuk menerima sesi pemain baru. Sesi game dapat diatur untuk menerima atau menolak semua sesi pemain baru.

Sintaks

```
FGameLiftGenericOutcome UpdatePlayerSessionCreationPolicy(EPlayerSessionCreationPolicy
policy)
```

Parameter

playerCreationSessionKebijakan

Nilai string yang menunjukkan apakah sesi game menerima pemain baru.

Nilai yang valid meliputi:

- `ACCEPT_ALL` — Menerima semua sesi pemain baru.
- `DENY_ALL` — Menolak semua sesi pemain baru.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Contoh ini menetapkan kebijakan bergabung sesi game saat ini untuk menerima semua pemain.

```
FGameLiftGenericOutcome outcome = gameLiftSdkModule-  
>UpdatePlayerSessionCreationPolicy(Aws::GameLift::Model::EPlayerSessionCreationPolicy::ACCEPT_A
```

GetGameSessionId()

Mengambil ID sesi permainan yang dihosting oleh proses server aktif.

Untuk proses idle yang tidak diaktifkan dengan sesi game, panggilan akan menampilkan file. [the section called “FGameLiftError”](#)

Sintaks

```
FGameLiftStringOutcome GetGameSessionId()
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai yang dikembalikan

Jika berhasil, ini mengembalikan ID sesi game sebagai objek [the section called “FGameLiftStringOutcome”](#). Jika tidak berhasil, mengembalikan pesan kesalahan.

Untuk proses idle yang tidak diaktifkan dengan sesi game, panggilan mengembalikan Success = True dan GameSessionId = "".

Contoh

```
//When a game session is created, Amazon GameLift Servers sends an activation request  
to the game server and passes along the game session object containing game properties  
and other settings.  
//Here is where a game server should take action based on the game session object.  
//Once the game server is ready to receive incoming player connections, it should  
invoke GameLiftServerAPI.ActivateGameSession()
```

```
auto onGameSession = [=](Aws::GameLift::Server::Model::GameSession gameSession)
{
    FString gameId = FString(gameSession.GetGameSessionId());
    UE_LOG(GameServerLog, Log, TEXT("GameSession Initializing: %s"), *gameId);
    gameLiftSdkModule->ActivateGameSession();
};
```

GetTerminationTime()

Mengembalikan waktu yang merupakan jadwal proses server akan ditutup, jika waktu penghentian tersedia. Proses server mengambil tindakan setelah menerima `onProcessTerminate()` panggilan balik dari Amazon GameLift Servers. Amazon GameLift Servers panggilan `onProcessTerminate()` untuk alasan berikut:

- Ketika proses server telah melaporkan kesehatan yang buruk atau belum Amazon GameLift Servers merespons.
- Saat mengakhiri instance selama acara scale-down.
- Ketika sebuah instance dihentikan karena gangguan [spot-instance](#).

Sintaks

```
AwsDateTimeOutcome GetTerminationTime()
```

Nilai yang ditampilkan

Jika berhasil, ini mengembalikan waktu penghentian sebagai objek `AwsDateTimeOutcome`. Nilainya adalah waktu penghentian, dinyatakan dalam kutu yang telah berlalu sejak. `000100:00:00` Misalnya, nilai waktu tanggal `2020-09-13 12:26:40 -000Z` sama dengan `637355968000000000` kutu. Jika tidak ada waktu penghentian tersedia, pesan kesalahan ditampilkan.

Jika proses belum menerima `ProcessParameters.OnProcessTerminate()` panggilan balik, pesan kesalahan akan dikembalikan. Untuk informasi selengkapnya tentang mematikan proses server, lihat [Menanggapi notifikasi shutdown proses server](#).

Contoh

```
AwsDateTimeOutcome TermTimeOutcome = gameLiftSdkModule->GetTerminationTime();
```

AcceptPlayerSession()

Memberi tahu Amazon GameLift Servers bahwa pemain dengan ID sesi pemain yang ditentukan telah terhubung ke proses server dan membutuhkan validasi. Amazon GameLift Servers memverifikasi bahwa ID sesi pemain valid. Setelah sesi pemain divalidasi, Amazon GameLift Servers mengubah status slot pemain dari RESERVED ke ACTIVE.

Sintaks

```
FGameLiftGenericOutcome AcceptPlayerSession(const FString& playerSessionId)
```

Parameter

playerSessionId

ID unik yang dikeluarkan oleh Amazon GameLift Servers saat sesi pemain baru dibuat.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Contoh ini menangani permintaan koneksi yang mencakup memvalidasi dan menolak sesi pemain yang tidak valid. IDs

```
bool GameLiftManager::AcceptPlayerSession(const FString& playerSessionId, const
FString& playerId)
{
    #if WITH_GAMELIFT
    UE_LOG(GameServerLog, Log, TEXT("Accepting GameLift PlayerSession: %s . PlayerId:
%s"), *playerSessionId, *playerId);
    FString gsId = GetCurrentGameSessionId();
    if (gsId.IsEmpty()) {
        UE_LOG(GameServerLog, Log, TEXT("No GameLift GameSessionId. Returning early!"));
        return false;
    }

    if (!gameLiftSdkModule->AcceptPlayerSession(playerSessionId).IsSuccess()) {
        UE_LOG(GameServerLog, Log, TEXT("PlayerSession not Accepted."));
    }
}
```

```
    return false;
}

// Add PlayerSession from internal data structures keeping track of connected players
connectedPlayerSessionIds.Add(playerSessionId);
idToPlayerSessionMap.Add(playerSessionId, PlayerSession{ playerId,
playerSessionId });
return true;
#else
return false;
#endif
}
```

RemovePlayerSession()

Memberitahu Amazon GameLift Servers bahwa pemain telah terputus dari proses server. Sebagai tanggapan, Amazon GameLift Servers ubah slot pemain menjadi tersedia.

Sintaks

```
FGameLiftGenericOutcome RemovePlayerSession(const FString& playerSessionId)
```

Parameter

playerSessionId

ID unik yang dikeluarkan oleh Amazon GameLift Servers saat sesi pemain baru dibuat.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

```
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()

Mengambil data sesi pemain yang mencakup pengaturan, metadata sesi, dan data pemain. Gunakan metode ini untuk mendapatkan informasi tentang hal-hal berikut:

- Sesi pemain tunggal
- Semua sesi pemain dalam sesi permainan
- Semua sesi pemain yang terkait dengan ID pemain tunggal

Sintaks

```
FGameLiftDescribePlayerSessionsOutcome DescribePlayerSessions(const
FGameLiftDescribePlayerSessionsRequest &describePlayerSessionsRequest)
```

Parameter

FGameLiftDescribePlayerSessionsRequest

the section called “FGameLiftDescribePlayerSessionsRequest” Objek yang menggambarkan sesi pemain mana yang akan diambil.

Nilai yang dikembalikan

Jika berhasil, ini mengembalikan objek [the section called “FGameLiftDescribePlayerSessionsOutcome”](#) yang berisi satu set objek sesi pemain yang sesuai dengan parameter permintaan.

Contoh

Contoh ini meminta semua sesi pemain yang terhubung secara aktif ke sesi permainan tertentu. Dengan menghilangkan NextToken dan mengatur nilai Batas ke 10, Amazon GameLift Servers mengembalikan catatan sesi 10 pemain pertama yang cocok dengan permintaan.

```
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()

Mengirim permintaan untuk menemukan pemain baru untuk slot terbuka dalam sesi permainan yang dibuat dengan FlexMatch. Untuk informasi selengkapnya, lihat [fitur FlexMatch isi ulang](#).

Tindakan ini asinkron. Jika pemain baru dicocokkan, kirimkan data Amazon GameLift Servers mak comblang yang diperbarui menggunakan fungsi panggilan balik. OnUpdateGameSession()

Proses server hanya dapat melakukan satu permintaan backfill match yang aktif dalam satu waktu. Untuk mengirim permintaan baru, panggil [StopMatchBackfill\(\)](#) terlebih dahulu untuk membatalkan permintaan asli.

Sintaks

```
FGameLiftStringOutcome StartMatchBackfill (FStartMatchBackfillRequest
&startBackfillRequest);
```

Parameter

[FStartMatchBackfillRequest](#)

StartMatchBackfillRequest Objek yang mengkomunikasikan informasi berikut:

- ID tiket untuk ditetapkan ke permintaan backfill. Informasi ini opsional; jika tidak ada ID yang diberikan, Amazon GameLift Servers akan menghasilkan satu.
- Matchmaker untuk dikirim permintaan. ARN konfigurasi penuh diperlukan. Nilai ini ada dalam data mak comblang sesi permainan.
- ID sesi permainan untuk mengisi ulang.
- Data perjodohan yang tersedia untuk pemain sesi permainan saat ini.

Nilai yang dikembalikan

Mengembalikan StartMatchBackfillOutcome objek dengan ID tiket isi ulang kecocokan, atau kegagalan dengan pesan kesalahan.

Contoh

```
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()

Membatalkan permintaan pengisian ulang pertandingan yang aktif. Untuk informasi selengkapnya, lihat [fitur FlexMatch isi ulang](#).

Sintaks

```
FGameLiftGenericOutcome StopMatchBackfill (FStopMatchBackfillRequest
&stopBackfillRequest);
```

Parameter

[FStopMatchBackfillRequest](#)

StopMatchBackfillRequest Objek yang mengidentifikasi tiket perjodohan untuk dibatalkan:

- ID tiket yang ditetapkan untuk permintaan pengisian ulang.
- Mak comblang permintaan isi ulang dikirim ke.
- Sesi permainan yang terkait dengan permintaan isi ulang.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

```
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()

Mengambil jalur ke sertifikat TLS yang digunakan untuk mengenkripsi koneksi jaringan antara sumber daya komputasi Amazon GameLift Servers Anywhere Anda dan Amazon GameLift Servers Anda dapat menggunakan jalur sertifikat saat mendaftarkan perangkat komputasi ke armada Amazon GameLift Servers Anywhere. Untuk informasi selengkapnya, lihat [RegisterCompute](#).

Sintaks

```
FGameLiftGetComputeCertificateOutcome FGameLiftServerSDKModule::GetComputeCertificate()
```

Nilai yang ditampilkan

Mengembalikan GetComputeCertificateResponse objek yang berisi berikut:

- CertificatePath: Jalur ke sertifikat TLS pada sumber daya komputasi Anda.
- HostName: Nama host sumber daya komputasi Anda.

Contoh

```

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());
    }
    #endif
}

```

```

    result.m_computeName = UTF8_TO_TCHAR(outres.GetComputeName());
    return FGameLiftGetComputeCertificateOutcome(result);
}
else {
    return FGameLiftGetComputeCertificateOutcome(FGameLiftError(outcome.GetError()));
}
#else
return FGameLiftGetComputeCertificateOutcome(FGameLiftGetComputeCertificateResult());
#endif
}

```

GetFleetRoleCredentials()

Mengambil kredensial peran IAM yang mengizinkan Amazon GameLift Servers untuk berinteraksi dengan orang lain. Layanan AWS Untuk informasi selengkapnya, lihat [Berkomunikasi dengan AWS sumber daya lain dari armada Anda](#).

Sintaks

```

FGameLiftGetFleetRoleCredentialsOutcome
FGameLiftServerSDKModule::GetFleetRoleCredentials(const
FGameLiftGetFleetRoleCredentialsRequest &request)

```

Parameter

[FGameLiftGetFleetRoleCredentialsRequest](#)

Nilai yang dikembalikan

Mengembalikan objek [the section called “FGameLiftGetFleetRoleCredentialsOutcome”](#).

Contoh

```

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
}
```

Server SDK untuk Amazon GameLift Servers versi 4 dan sebelumnya

Referensi ini mendokumentasikan SDK server untuk Amazon GameLift Servers, versi 4.x dan sebelumnya. Server SDK menyediakan fungsionalitas inti yang digunakan server game Anda untuk berkomunikasi dengan Amazon GameLift Servers layanan. Misalnya, server game Anda menerima permintaan dari layanan untuk memulai dan menghentikan sesi permainan dan menyediakan pembaruan status sesi permainan reguler ke layanan. Integrasikan server game Anda dengan SDK server sebelum Anda menerapkannya untuk hosting.

Gunakan referensi SDK server ini untuk mengintegrasikan server game multipemain kustom Anda untuk hosting Amazon GameLift Servers. Untuk panduan tentang proses integrasi, lihat [Tambahkan Amazon GameLift Servers ke server game Anda](#).

Versi utama terbaru dari SDK server untuk Amazon GameLift Servers adalah 5.x. Fitur hosting berikut memerlukan pembaruan ke versi 5.x:

- Amazon GameLift Servers Dimanapun

- Amazon GameLift Servers kontainer terkelola
- Amazon GameLift Servers plugin untuk Unreal Engine dan Unity

Topik

- [SDK server C ++ untuk Amazon GameLift Servers 4.x -- Tindakan](#)
- [SDK server C # untuk Amazon GameLift Servers 4.x -- Tindakan](#)
- [Server SDK \(Unreal\) untuk Amazon GameLift Servers -- Tindakan](#)

SDK server C ++ untuk Amazon GameLift Servers 4.x -- Tindakan

Gunakan referensi SDK server untuk mengintegrasikan game multipemain Anda untuk hosting. Amazon GameLift Servers Untuk panduan tentang proses integrasi, lihat [Tambahkan Amazon GameLift Servers ke server game Anda](#).

Note

Referensi ini untuk versi SDK server yang lebih lama untuk Amazon GameLift Servers. Untuk versi terbaru, lihat [Server C ++ SDK 5.x untuk Amazon GameLift Servers -- Tindakan](#).

SDK server C ++ untuk Amazon GameLift Servers 4.x - Tipe data

Gunakan referensi SDK server untuk mengintegrasikan game multipemain Anda untuk hosting Amazon GameLift Servers. Untuk panduan tentang proses integrasi, lihat [Tambahkan Amazon GameLift Servers ke server game Anda](#).

Note

Referensi ini untuk versi SDK server yang lebih lama Amazon GameLift Servers. Untuk versi terbaru, lihat [Server C ++ SDK 5.x untuk Amazon GameLift Servers -- Tipe data](#).

API ini didefinisikan dalam `GameLiftServerAPI.h`, `LogParameters.h`, dan `ProcessParameters.h`.

[SDK server C ++ untuk Amazon GameLift Servers 4.x -- Tindakan](#)

DescribePlayerSessionsRequest

Jenis data ini digunakan untuk menentukan sesi pemain untuk diambil. Anda bisa menggunakannya sebagai berikut:

- Menyediakan `PlayerSessionId` untuk meminta sesi pemain tertentu.
- Berikan permintaan `GameSessionId` untuk semua sesi pemain dalam sesi permainan yang ditentukan.
- Berikan permintaan `PlayerId` untuk semua sesi pemain untuk pemain yang ditentukan.

Untuk koleksi sesi pemain yang besar, gunakan parameter pemberian nomor halaman untuk mengambil hasil dalam blok berurutan.

Daftar Isi

GameSessionId

Pengidentifikasi sesi game yang unik. Gunakan parameter ini untuk meminta semua sesi pemain untuk sesi game yang ditentukan. Format ID sesi game adalah sebagai berikut: `arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string>`. Nilai `<ID string>` adalah string ID kustom atau (jika salah satu ditentukan saat sesi game dibuat) string yang dihasilkan.

Tipe: String

Wajib: Tidak

Kuota

Jumlah hasil maksimum yang akan dikembalikan. Gunakan parameter ini dengan `NextToken` untuk mendapatkan hasil sebagai satu set halaman berurutan. Jika ID sesi pemain ditentukan, parameter ini diabaikan.

Jenis: Integer

Wajib: Tidak

NextToken

Token yang menunjukkan awal dari halaman berurutan berikutnya dari hasil. Gunakan token yang dikembalikan dengan panggilan sebelumnya untuk tindakan ini. Untuk menentukan awal set hasil, jangan menentukan nilai. Jika ID sesi pemain ditentukan, parameter ini diabaikan.

Tipe: String

Wajib: Tidak

PlayerId

Pengidentifikasi unik untuk pemain. Pemain IDs ditentukan oleh pengembang. Lihat [Hasilkan pemain IDs](#).

Tipe: String

Wajib: Tidak

PlayerSessionId

Pengidentifikasi unik untuk sesi pemain.

Tipe: String

Wajib: Tidak

PlayerSessionStatusFilter

Status sesi pemain untuk mem-filter hasil. Kemungkinan status sesi pemain meliputi:

- RESERVED — Permintaan sesi pemain telah diterima, namun pemain belum terhubung ke proses server dan/atau divalidasi.
- ACTIVE — Pemain telah divalidasi oleh proses server dan saat ini terhubung.
- COMPLETED — Sambungan pemain telah terputus.
- TIMEDOUT — Permintaan sesi pemain diterima, namun pemain tidak terhubung dan/atau tidak divalidasi dalam batas waktu (60 detik).

Tipe: String

Wajib: Tidak

LogParameters

Tipe data ini digunakan untuk mengidentifikasi file mana yang dihasilkan selama sesi permainan yang Anda inginkan Amazon GameLift Servers untuk mengunggah dan menyimpan setelah sesi permainan berakhir. Informasi ini dikomunikasikan kepada Amazon GameLift Servers layanan dalam [ProcessReady\(\)](#) panggilan.

Daftar Isi

logPaths

Jalur direktori ke file log server game yang Anda inginkan Amazon GameLift Servers untuk menyimpan untuk akses masa depan. File-file ini dihasilkan selama setiap sesi game. Jalur file dan nama didefinisikan dalam server game Anda dan disimpan dalam direktori build root game. Jalur log harus absolut. Misalnya, jika build game Anda menyimpan log sesi game di jalur seperti `MyGame\sessionlogs\`, maka jalur log akan menjadi `c:\game\MyGame\sessionLogs` (di instans Windows) atau `/local/game/MyGame/sessionLogs` (di instans Linux).

Jenis: `std::vector<std::string>`

Wajib: Tidak

ProcessParameters

Tipe data ini berisi kumpulan parameter yang dikirim ke Amazon GameLift Servers layanan dalam [ProcessReady\(\)](#) panggilan.

Daftar Isi

port

Nomor port yang akan didengarkan proses server untuk koneksi pemain baru. Nilai harus masuk ke dalam kisaran port yang dikonfigurasi untuk setiap armada yang men-deploy build server game ini. Nomor port ini termasuk dalam sesi game dan objek sesi pemain, yang digunakan sesi game saat menghubungkan ke proses server.

Jenis: Integer

Wajib: Ya

logParameters

Obyek dengan daftar jalur direktori untuk file log sesi game.

Jenis: `Aws::GameLift::Server::` [LogParameters](#)

Wajib: Tidak

onStartGameSesi

Nama fungsi callback yang Amazon GameLift Servers layanan memanggil untuk mengaktifkan sesi permainan baru. Amazon GameLift Servers memanggil fungsi ini sebagai tanggapan atas permintaan klien [CreateGameSession](#). Fungsi callback melewati [GameSession](#) objek (didefinisikan dalam Amazon GameLift Servers Referensi API Layanan).

Tipe: `const std::function<void(Aws::GameLift::Model::GameSession)>
onStartGameSession`

Wajib: Ya

onProcessTerminate

Nama fungsi callback yang Amazon GameLift Servers layanan memanggil untuk memaksa proses server dimatikan. Setelah memanggil fungsi ini, Amazon GameLift Servers menunggu lima menit hingga proses server dimatikan dan merespons dengan [ProcessEnding\(\)](#) panggilan. Jika tidak ada respon yang diterima, proses server akan dimatikan.

Tipe: `std::function<void()> onProcessTerminate`

Wajib: Tidak

onHealthCheck

Nama fungsi callback yang Amazon GameLift Servers layanan memanggil untuk meminta laporan status kesehatan dari proses server. Amazon GameLift Servers memanggil fungsi ini setiap 60 detik. Setelah memanggil fungsi ini Amazon GameLift Servers menunggu 60 detik untuk respons, dan jika tidak ada yang diterima, mencatat proses server sebagai tidak sehat.

Tipe: `std::function<bool()> onHealthCheck`

Wajib: Tidak

onUpdateGameSesi

Nama fungsi callback yang Amazon GameLift Servers layanan memanggil untuk meneruskan objek sesi game yang diperbarui ke proses server. Amazon GameLift Servers memanggil fungsi ini ketika permintaan [pengisian ulang kecocokan](#) telah diproses untuk memberikan data mak comblang yang diperbarui. Ini melewati [GameSession](#) objek, pembaruan status (`updateReason`), dan ID tiket penimbunan ulang pertandingan.

```
Tipe: std::function<void(Aws::GameLift::Server::Model::UpdateGameSession)>  
onUpdateGameSession
```

Wajib: Tidak

StartMatchBackfillRequest

Jenis data ini digunakan untuk mengirim permintaan backfill matchmaking. Informasi tersebut dikomunikasikan kepada Amazon GameLift Servers layanan dalam [StartMatchBackfill\(\)](#) panggilan.

Daftar Isi

GameSessionArn

Pengidentifikasi sesi game yang unik. Tindakan API [GetGameSessionId\(\)](#) mengembalikan pengidentifikasi dalam format ARN.

Tipe: String

Diperlukan: Ya

MatchmakingConfigurationArn

Pengidentifikasi unik, dalam bentuk ARN, yang akan digunakan matchmaker untuk permintaan ini. Untuk menemukan matchmaker yang digunakan untuk membuat sesi game asli, lihat di objek sesi game, di properti data matchmaker. Pelajari selengkapnya tentang data matchmaker di [Bekerja dengan data matchmaker](#).

Tipe: String

Diperlukan: Ya

Pemain

Satu set data yang mewakili semua pemain yang saat ini dalam sesi game. Matchmaker menggunakan informasi ini untuk mencari pemain baru yang cocok untuk pemain saat ini. Lihat Amazon GameLift Servers Panduan Referensi API untuk deskripsi format objek Player. Untuk menemukan atribut pemain IDs, dan tugas tim, lihat objek sesi permainan, di properti data mak comblang. Jika latensi digunakan oleh matchmaker, kumpulkan latensi yang diperbarui untuk wilayah saat ini dan sertakan dalam data masing-masing pemain.

Jenis: std: vektorhttps://docs.aws.amazon.com/gamelift/latest/apireference/API_Player.html
<player>

Wajib: Ya

TicketId

Pengidentifikasi unik untuk tiket permintaan matchmaking atau backfill match. Jika tidak ada nilai yang diberikan di sini, Amazon GameLift Servers akan menghasilkan satu dalam bentuk UUID. Gunakan pengidentifikasi ini untuk melacak status tiket backfill match atau membatalkan permintaan jika diperlukan.

Tipe: String

Wajib: Tidak

StopMatchBackfillRequest

Jenis data ini digunakan untuk membatalkan permintaan backfill matchmaking. Informasi tersebut dikomunikasikan kepada Amazon GameLift Servers layanan dalam [StopMatchBackfill\(\)](#) panggilan.

Daftar Isi

GameSessionArn

Pengidentifikasi sesi game unik yang terkait dengan permintaan yang dibatalkan.

Tipe: String

Diperlukan: Ya

MatchmakingConfigurationArn

Pengidentifikasi unik dari matchmaker sebagai tujuan pengiriman permintaan ini.

Tipe: String

Diperlukan: Ya

TicketId

Pengidentifikasi unik dari tiket backfill match yang akan dibatalkan.

Tipe: String

Diperlukan: Ya

SDK server C ++ untuk Amazon GameLift Servers 4.x - Tipe data

Topik

- [AcceptPlayerSession\(\)](#)
- [ActivateGameSession\(\)](#)
- [DescribePlayerSessions\(\)](#)
- [GetGameSessionId\(\)](#)
- [GetInstanceCertificate\(\)](#)
- [GetSdkVersion\(\)](#)
- [GetTerminationTime\(\)](#)
- [InitSDK\(\)](#)
- [ProcessEnding\(\)](#)
- [ProcessReady\(\)](#)
- [ProcessReadyAsync\(\)](#)
- [RemovePlayerSession\(\)](#)
- [StartMatchBackfill\(\)](#)
- [StopMatchBackfill\(\)](#)
- [TerminateGameSession\(\)](#)
- [UpdatePlayerSessionCreationPolicy\(\)](#)
- [Hancurkan \(\)](#)

AcceptPlayerSession()

Memberitahu Amazon GameLift Servers layanan bahwa pemain dengan ID sesi pemain tertentu telah terhubung ke proses server dan perlu validasi. Amazon GameLift Serversmemverifikasi bahwa ID sesi pemain valid—yaitu, ID pemain telah memesan slot pemain dalam sesi permainan. Setelah divalidasi, Amazon GameLift Servers ubah status slot pemain dari RESERVED ke ACTIVE.

Sintaks

```
GenericOutcome AcceptPlayerSession(const std::string& playerSessionId);
```


Parameter

playerSessionId

ID unik yang dikeluarkan oleh Amazon GameLift Servers layanan sebagai respons terhadap panggilan ke tindakan AWS [CreatePlayerSession](#) SDK Amazon GameLift Servers API. Client game mereferensi ID ini saat menghubungkan ke proses server.

Jenis: `std::string`

Wajib: Ya

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Contoh ini menggambarkan fungsi untuk menangani permintaan koneksi, termasuk memvalidasi dan menolak sesi pemain yang tidak valid. 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()

Memberi tahu Amazon GameLift Servers layanan bahwa proses server telah memulai sesi permainan dan sekarang siap menerima koneksi pemain. Tindakan ini harus dipanggil sebagai bagian dari fungsi callback `onStartGameSession()`, setelah semua inisialisasi sesi game selesai.

Sintaks

```
GenericOutcome ActivateGameSession();
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Contoh ini menunjukkan `ActivateGameSession()` yang dipanggil sebagai bagian dari fungsi callback `onStartGameSession()`.

```
void onStartGameSession(Aws::GameLift::Model::GameSession myGameSession)
{
    // game-specific tasks when starting a new game session, such as loading map
    GenericOutcome outcome = Aws::GameLift::Server::ActivateGameSession();
}
```

DescribePlayerSessions()

Mengambil data sesi pemain, termasuk pengaturan, metadata sesi, dan data pemain. Gunakan tindakan ini untuk mendapatkan informasi untuk satu sesi pemain, untuk semua sesi pemain dalam sesi game, atau untuk semua sesi pemain yang terkait dengan ID pemain tunggal.

Sintaks

```
DescribePlayerSessionsOutcome DescribePlayerSessions (
    const Aws::GameLift::Server::Model::DescribePlayerSessionsRequest
    &describePlayerSessionsRequest);
```

Parameter

describePlayerSessionsPermintaan

Sebuah objek [DescribePlayerSessionsRequest](#) yang menjelaskan sesi pemain mana yang diambil.

Wajib: Ya

Nilai yang dikembalikan

Jika berhasil, ini mengembalikan objek `DescribePlayerSessionsOutcome` yang berisi satu set objek sesi pemain yang sesuai dengan parameter permintaan. Objek sesi pemain memiliki struktur yang identik dengan tipe [PlayerSession](#) data AWS SDK Amazon GameLift Servers API.

Contoh

Contoh ini menggambarkan permintaan untuk semua sesi pemain yang secara aktif terhubung ke sesi game tertentu. Dengan menghilangkan `NextToken` dan menyetel `Limit` nilai ke 10, Amazon GameLift Servers mengembalikan catatan sesi 10 pemain pertama yang cocok dengan permintaan.

```
// 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()

Mengambil pengidentifikasi unik untuk sesi game yang saat ini sedang di-host oleh proses server, jika proses server aktif. Pengidentifikasi dikembalikan dalam format ARN: `arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string>`.

Untuk proses idle yang belum diaktifkan dengan sesi permainan, panggilan mengembalikan `Success = True` dan `GameSessionId = ""` (string kosong).

Sintaks

```
AwsStringOutcome GetGameSessionId();
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai yang dikembalikan

Jika berhasil, ini mengembalikan ID sesi game sebagai objek `AwsStringOutcome`. Jika tidak berhasil, ini mengembalikan pesan kesalahan.

Contoh

```
Aws::GameLift::AwsStringOutcome sessionIdOutcome =  
    Aws::GameLift::Server::GetGameSessionId();
```

GetInstanceCertificate()

Mengambil lokasi file sertifikat TLS yang dikodekan pem-encode yang terkait dengan armada dan instance-nya. AWS Certificate Manager menghasilkan sertifikat ini saat Anda membuat armada baru dengan konfigurasi sertifikat yang disetel ke `GENERATED`. Gunakan sertifikat ini untuk membuat koneksi yang aman dengan client game dan untuk mengenkripsi komunikasi client/server.

Sintaks

```
GetInstanceCertificateOutcome GetInstanceCertificate();
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai yang dikembalikan

Jika berhasil, mengembalikan `GetInstanceCertificateOutcome` objek yang berisi lokasi file sertifikat TLS armada dan rantai sertifikat, yang disimpan pada instance. File sertifikat root, diekstrak dari rantai sertifikat, juga disimpan pada instance. Jika tidak berhasil, ini mengembalikan pesan kesalahan.

Untuk informasi selengkapnya tentang sertifikat dan data rantai sertifikat, lihat [Elemen GetCertificate Respons](#) di Referensi AWS Certificate Manager API.

Contoh

```
Aws::GameLift::GetInstanceCertificateOutcome certificateOutcome =  
    Aws::GameLift::Server::GetInstanceCertificate();
```

GetSdkVersion()

Mengembalikan nomor versi dari SDK yang digunakan saat ini.

Sintaks

```
AwsStringOutcome GetSdkVersion();
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai yang dikembalikan

Jika berhasil, ini mengembalikan versi SDK saat ini sebagai objek `AwsStringOutcome`. String yang dikembalikan mencakup nomor versi saja (mis. "3.1.5"). Jika tidak berhasil, ini mengembalikan pesan kesalahan.

Contoh

```
Aws::GameLift::AwsStringOutcome SdkVersionOutcome =  
    Aws::GameLift::Server::GetSdkVersion();
```

GetTerminationTime()

Mengembalikan waktu yang merupakan jadwal proses server akan ditutup, jika waktu penghentian tersedia. Proses server mengambil tindakan ini setelah menerima `onProcessTerminate()` panggilan balik dari Amazon GameLift Servers layanan. Amazon GameLift Servers [dapat menelepon `onProcessTerminate\(\)`](#) karena alasan berikut: (1) ketika proses server melaporkan kesehatan yang buruk atau belum merespons Amazon GameLift Servers, (2) saat menghentikan instance selama peristiwa penurunan skala, atau (3) saat instance dihentikan karena gangguan Spot.

Jika proses telah menerima callback `onProcessTerminate()`, nilai yang dikembalikan adalah perkiraan waktu penghentian. Jika proses belum menerima callback `onProcessTerminate()`, pesan kesalahan dikembalikan. Pelajari selengkapnya tentang [mematikan proses server](#).

Sintaks

```
AwsLongOutcome GetTerminationTime();
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai yang dikembalikan

Jika berhasil, ini mengembalikan waktu penghentian sebagai objek `AwsLongOutcome`. Nilainya adalah waktu penghentian, dinyatakan dalam detik yang telah berlalu sejak 0001 00:00:00. Misalnya, nilai waktu tanggal 2020-09-13 12:26:40 -000Z sama dengan 637355968000000000 detik. Jika tidak ada waktu penghentian tersedia, pesan kesalahan ditampilkan.

Contoh

```
Aws::GameLift::AwsLongOutcome TermTimeOutcome =  
    Aws::GameLift::Server::GetTerminationTime();
```

InitSDK()

Menginisialisasi Amazon GameLift Servers SDK. Metode ini harus dipanggil saat peluncuran, sebelum inisialisasi Amazon GameLift Servers terkait lainnya terjadi.

Sintaks

```
InitSDKOutcome InitSDK();
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai yang dikembalikan

Jika berhasil, mengembalikan `InitSdkOutcome` objek yang menunjukkan bahwa proses server siap untuk memanggil [ProcessReady\(\)](#).

Contoh

```
Aws::GameLift::Server::InitSDKOutcome initOutcome =  
    Aws::GameLift::Server::InitSDK();
```

ProcessEnding()

Memberitahu Amazon GameLift Servers layanan bahwa proses server dimatikan. Metode ini harus dipanggil setelah semua tugas pembersihan lainnya, termasuk mematikan semua sesi game aktif. Metode ini harus keluar dengan kode keluar 0; sebuah kode non-nol menghasilkan pesan kejadian bahwa proses tidak keluar dengan bersih.

Setelah metode keluar dengan kode 0, Anda dapat mengakhiri proses dengan kode keluar yang sukses. Anda juga dapat keluar dari proses dengan kode kesalahan. Jika Anda keluar dengan kode kesalahan, kejadian armada akan menunjukkan proses yang dihentikan secara tidak normal (SERVER_PROCESS_TERMINATED_UNHEALTHY).

Sintaks

```
GenericOutcome ProcessEnding();
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

```
Aws::GameLift::GenericOutcome outcome = Aws::GameLift::Server::ProcessEnding();  
if (outcome.Success)  
    exit(0); // exit with success  
// otherwise, exit with error code  
exit(errorCode);
```

ProcessReady()

Memberitahu Amazon GameLift Servers layanan bahwa proses server siap untuk meng-host sesi permainan. Panggil metode ini setelah berhasil meminta [InitSDK\(\)](#) dan menyelesaikan tugas persiapan yang diperlukan sebelum proses server dapat menjadi host sesi game. Metode ini harus dipanggil hanya satu kali per proses.

Panggilan ini sinkron. Untuk membuat panggilan asinkron, gunakan [ProcessReadyAsync\(\)](#). Lihat [Inisialisasi proses server](#) untuk detail selengkapnya.

Sintaks

```
GenericOutcome ProcessReady(  
    const Aws::GameLift::Server::ProcessParameters &processParameters);
```

Parameter

processParameters

Sebuah objek [ProcessParameters](#) yang mengomunikasikan informasi berikut tentang proses server:

- Nama metode callback, diimplementasikan dalam kode server game, yang dipanggil Amazon GameLift Servers layanan untuk berkomunikasi dengan proses server.
- Nomor port yang didengarkan oleh proses server.
- Jalur ke file khusus sesi game apa pun yang Amazon GameLift Servers ingin Anda tangkap dan simpan.

Wajib: Ya

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Contoh ini menggambarkan panggilan [ProcessReady\(\)](#) dan implementasi fungsi callback.

```
// 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()

Memberitahu Amazon GameLift Servers layanan bahwa proses server siap untuk meng-host sesi permainan. Metode ini harus dipanggil setelah proses server siap untuk menjadi host sesi game. Parameter menentukan nama fungsi callback Amazon GameLift Servers untuk dipanggil dalam keadaan tertentu. Kode server game harus menerapkan fungsi-fungsi ini.

Ini adalah panggilan asinkron. Untuk membuat panggilan sinkron, gunakan [ProcessReady\(\)](#). Lihat [Inisialisasi proses server](#) untuk detail selengkapnya.

Sintaks

```
GenericOutcomeCallable ProcessReadyAsync(  
    const Aws::GameLift::Server::ProcessParameters &processParameters);
```

Parameter

processParameters

Sebuah objek [ProcessParameters](#) yang mengomunikasikan informasi berikut tentang proses server:

- Nama metode callback, diimplementasikan dalam kode server game, yang dipanggil Amazon GameLift Servers layanan untuk berkomunikasi dengan proses server.
- Nomor port yang didengarkan oleh proses server.
- Jalur ke file khusus sesi game apa pun yang Amazon GameLift Servers ingin Anda tangkap dan simpan.

Wajib: Ya

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

```
// 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()

Memberi tahu Amazon GameLift Servers layanan bahwa pemain dengan ID sesi pemain yang ditentukan telah terputus dari proses server. Sebagai tanggapan, Amazon GameLift Servers mengubah slot pemain menjadi tersedia, yang memungkinkannya ditugaskan ke pemain baru.

Sintaks

```
GenericOutcome RemovePlayerSession(
    const std::string& playerId);
```

Parameter

playerSessionId

ID unik yang dikeluarkan oleh Amazon GameLift Servers layanan sebagai respons terhadap panggilan ke tindakan AWS [CreatePlayerSession](#) SDK Amazon GameLift Servers API. Client game mereferensi ID ini saat menghubungkan ke proses server.

Jenis: std::string

Wajib: Ya

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

```
Aws::GameLift::GenericOutcome disconnectOutcome =  
    Aws::GameLift::Server::RemovePlayerSession(playerSessionId);
```

StartMatchBackfill()

Mengirim permintaan untuk menemukan pemain baru untuk slot terbuka dalam sesi permainan yang dibuat dengan FlexMatch. Lihat juga aksi AWS SDK [StartMatchBackfill\(\)](#). Dengan tindakan ini, permintaan backfill match dapat dimulai dengan proses server game yang menjadi host sesi game. Pelajari lebih lanjut tentang [fitur FlexMatch isi ulang](#).

Tindakan ini asinkron. Jika pemain baru berhasil dicocokkan, Amazon GameLift Servers layanan akan mengirimkan data mak comblang yang diperbarui dengan menjalankan fungsi panggilan balik. `OnUpdateGameSession()`

Proses server hanya dapat melakukan satu permintaan backfill match yang aktif dalam satu waktu. Untuk mengirim permintaan baru, panggil [StopMatchBackfill\(\)](#) terlebih dahulu untuk membatalkan permintaan asli.

Sintaks

```
StartMatchBackfillOutcome StartMatchBackfill (
```

```
const Aws::GameLift::Server::Model::StartMatchBackfillRequest
&startBackfillRequest);
```

Parameter

StartMatchBackfillRequest

Sebuah objek [StartMatchBackfillRequest](#) yang mengkomunikasikan informasi berikut:

- ID tiket untuk ditetapkan ke permintaan backfill. Informasi ini opsional; jika tidak ada ID yang diberikan, Amazon GameLift Servers akan dibuat secara otomatis.
- Matchmaker untuk dikirim permintaan. ARN konfigurasi penuh diperlukan. Nilai ini dapat diperoleh dari data matchmaker sesi game.
- ID dari sesi game yang sedang di-backfill.
- Data matchmaking yang tersedia untuk pemain sesi game saat ini.

Wajib: Ya

Nilai yang dikembalikan

Mengembalikan StartMatchBackfillOutcome objek dengan tiket isi ulang pertandingan atau kegagalan dengan pesan kesalahan. Status tiket dapat dilacak menggunakan tindakan AWS SDK [DescribeMatchmaking\(\)](#).

Contoh

```
// 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()

Membatalkan permintaan backfill match aktif yang dibuat dengan [StartMatchBackfill\(\)](#). Lihat juga aksi AWS SDK [StopMatchmaking\(\)](#). Pelajari lebih lanjut tentang [fitur FlexMatch isi ulang](#).

Sintaks

```
GenericOutcome StopMatchBackfill (
    const Aws::GameLift::Server::Model::StopMatchBackfillRequest &stopBackfillRequest);
```

Parameter

StopMatchBackfillRequest

Sebuah objek [StopMatchBackfillRequest](#) yang mengidentifikasi tiket matchmaking untuk membatalkan:

- ID tiket yang ditetapkan ke permintaan backfill yang dibatalkan
- matchmaker yang dikirim permintaan backfill
- sesi game yang terkait dengan permintaan backfill

Wajib: Ya

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

```
// 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()

Metode ini tidak lagi digunakan dengan versi 4.0.1. Sebaliknya, proses server harus memanggil [ProcessEnding\(\)](#) setelah sesi permainan berakhir.

Memberitahu Amazon GameLift Servers layanan bahwa proses server telah mengakhiri sesi permainan saat ini. Tindakan ini dipanggil ketika proses server akan tetap aktif dan siap untuk menjadi host sesi game baru. Ini harus dipanggil hanya setelah prosedur penghentian sesi permainan Anda selesai, karena ini menandakan Amazon GameLift Servers bahwa proses server segera tersedia untuk meng-host sesi permainan baru.

Tindakan ini tidak dipanggil jika proses server akan dimatikan setelah sesi game berhenti. Sebagai gantinya, panggil [ProcessEnding\(\)](#) untuk memberi tahu bahwa sesi game dan proses server akan berakhir.

Sintaks

```
GenericOutcome TerminateGameSession();
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

UpdatePlayerSessionCreationPolicy()

Memperbarui kemampuan sesi game saat ini untuk menerima sesi pemain baru. Sesi game dapat diatur untuk menerima atau menolak semua sesi pemain baru. Lihat juga aksi AWS SDK [UpdateGameSession\(\)](#).

Sintaks

```
GenericOutcome UpdatePlayerSessionCreationPolicy(  
    Aws::GameLift::Model::PlayerSessionCreationPolicy newPlayerSessionPolicy);
```

Parameter

newPlayerSessionKebijakan

Nilai string yang menunjukkan apakah sesi game menerima pemain baru.

Jenis: `Aws::GameLift::Model::PlayerSessionCreationPolicy` enum. Nilai yang valid meliputi:

- `ACCEPT_ALL` — Menerima semua sesi pemain baru.
- `DENY_ALL` — Menolak semua sesi pemain baru.

Wajib: Ya

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Contoh ini menetapkan kebijakan bergabung sesi game saat ini untuk menerima semua pemain.

```
Aws::GameLift::GenericOutcome outcome =  
    Aws::GameLift::Server::UpdatePlayerSessionCreationPolicy(Aws::GameLift::Model::PlayerSessionCr
```

Hancurkan ()

Membersihkan memori yang dialokasikan oleh `initSDK ()` selama inisialisasi server game. Gunakan metode ini setelah Anda mengakhiri proses server game untuk menghindari pemborosan memori server.

Sintaks

```
GenericOutcome Aws::GameLift::Server::Destroy();
```


Parameter

Tidak ada parameter.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Contoh ini membersihkan memori yang dialokasikan oleh initSDK setelah proses server game berakhir.

```
if (Aws::GameLift::Server::ProcessEnding().IsSuccess()) {  
    Aws::GameLift::Server::Destroy();  
    exit(0);  
}
```

SDK server C # untuk Amazon GameLift Servers 4.x -- Tindakan

Gunakan referensi SDK server untuk mengintegrasikan game multipemain Anda untuk hosting Amazon GameLift Servers. Untuk panduan tentang proses integrasi, lihat [Tambahkan Amazon GameLift Servers ke server game Anda](#).

Note

Referensi ini untuk versi SDK server yang lebih lama untuk Amazon GameLift Servers. Untuk versi terbaru, lihat [C# server SDK 5.x untuk Amazon GameLift Servers -- Tindakan](#).

C# server SDK untuk Amazon GameLift Servers 4.x - Jenis data

Gunakan referensi SDK server untuk mengintegrasikan game multipemain Anda untuk hosting Amazon GameLift Servers. Untuk panduan tentang proses integrasi, lihat [Tambahkan Amazon GameLift Servers ke server game Anda](#).

 Note

Referensi ini untuk versi SDK server yang lebih lama untuk Amazon GameLift Servers. Untuk versi terbaru, lihat [C# server SDK 5.x untuk Amazon GameLift Servers - Tipe data](#).

SDK server C # untuk Amazon GameLift Servers 4.x -- Tindakan

Topik

- [LogParameters](#)
- [DescribePlayerSessionsRequest](#)
- [ProcessParameters](#)
- [StopMatchBackfillRequest](#)

LogParameters

Tipe data ini digunakan untuk mengidentifikasi file mana yang dihasilkan selama sesi permainan yang Anda inginkan Amazon GameLift Servers untuk mengunggah dan menyimpan setelah sesi permainan berakhir. Informasi ini dikomunikasikan kepada Amazon GameLift Servers layanan dalam [ProcessReady\(\)](#) panggilan.

Daftar Isi

logPaths

Daftar jalur direktori ke file log server game yang Anda inginkan Amazon GameLift Servers untuk menyimpan untuk akses masa depan. File-file ini dihasilkan oleh proses server selama setiap sesi game; jalur file dan nama didefinisikan dalam server game Anda dan disimpan dalam direktori build root game. Jalur log harus absolut. Misalnya, jika build game Anda menyimpan log sesi game di jalur seperti MyGame\sessionlogs\, maka jalur log akan menjadi c:\game\MyGame\sessionLogs (di instans Windows) atau /local/game/MyGame/sessionLogs (di instans Linux).

Jenis: Daftar<String>

Wajib: Tidak

DescribePlayerSessionsRequest

Jenis data ini digunakan untuk menentukan sesi pemain untuk diambil. Ini dapat digunakan dalam beberapa cara: (1) menyediakan `PlayerSessionId` untuk meminta sesi pemain tertentu; (2) menyediakan `GameSessionId` untuk meminta semua sesi pemain dalam sesi permainan yang ditentukan; atau (3) menyediakan `PlayerId` untuk meminta semua sesi pemain untuk pemain yang ditentukan. Untuk koleksi sesi pemain yang besar, gunakan parameter pemberian nomor halaman untuk mengambil hasil dalam halaman berurutan.

Daftar Isi

GameSessionId

Pengidentifikasi sesi game yang unik. Gunakan parameter ini untuk meminta semua sesi pemain untuk sesi game yang ditentukan. Format ID sesi game adalah sebagai berikut:
`arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string>`. Nilai `<ID string>` adalah string ID kustom (jika salah satu ditentukan saat sesi game dibuat) atau string yang dihasilkan.

Tipe: String

Wajib: Tidak

Kuota

Jumlah hasil maksimum yang akan dikembalikan. Gunakan parameter ini dengan `NextToken` untuk mendapatkan hasil sebagai satu set halaman berurutan. Jika ID sesi pemain ditentukan, parameter ini diabaikan.

Jenis: Integer

Wajib: Tidak

NextToken

Token yang menunjukkan awal dari halaman berurutan berikutnya dari hasil. Gunakan token yang dikembalikan dengan panggilan sebelumnya untuk tindakan ini. Untuk menentukan awal set hasil, jangan menentukan nilai. Jika ID sesi pemain ditentukan, parameter ini diabaikan.

Tipe: String

Wajib: Tidak

PlayerId

Pengidentifikasi unik untuk pemain. Pemain IDs ditentukan oleh pengembang. Lihat [Hasilkan pemain IDs](#).

Tipe: String

Wajib: Tidak

PlayerSessionId

Pengidentifikasi unik untuk sesi pemain.

Tipe: String

Wajib: Tidak

PlayerSessionStatusFilter

Status sesi pemain untuk mem-filter hasil. Kemungkinan status sesi pemain meliputi:

- RESERVED — Permintaan sesi pemain telah diterima, namun pemain belum terhubung ke proses server dan/atau divalidasi.
- ACTIVE — Pemain telah divalidasi oleh proses server dan saat ini terhubung.
- COMPLETED — Sambungan pemain telah terputus.
- TIMEDOUT — Permintaan sesi pemain diterima, namun pemain tidak terhubung dan/atau tidak divalidasi dalam batas waktu (60 detik).

Tipe: String

Wajib: Tidak

ProcessParameters

Tipe data ini berisi kumpulan parameter yang dikirim ke Amazon GameLift Servers layanan dalam [ProcessReady\(\)](#) panggilan.

Daftar Isi

port

Nomor port yang akan didengarkan proses server untuk koneksi pemain baru. Nilai harus masuk ke dalam kisaran port yang dikonfigurasi untuk setiap armada yang men-deploy build server game

ini. Nomor port ini termasuk dalam sesi game dan objek sesi pemain, yang digunakan sesi game saat menghubungkan ke proses server.

Jenis: Integer

Wajib: Ya

logParameters

Obyek dengan daftar jalur direktori untuk file log sesi game.

Tipe: `Aws::GameLift::Server::LogParameters`

Wajib: Ya

onStartGameSesi

Nama fungsi callback yang Amazon GameLift Servers layanan memanggil untuk mengaktifkan sesi permainan baru. Amazon GameLift Servers memanggil fungsi ini sebagai tanggapan atas permintaan klien [CreateGameSession](#). Fungsi callback mengambil [GameSession](#) objek (didefinisikan dalam Amazon GameLift Servers Referensi API Layanan).

Tipe: `void OnStartGameSessionDelegate(GameSession gameSession)`

Wajib: Ya

onProcessTerminate

Nama fungsi callback yang Amazon GameLift Servers layanan memanggil untuk memaksa proses server dimatikan. Setelah memanggil fungsi ini, Amazon GameLift Servers menunggu lima menit hingga proses server dimatikan dan merespons dengan [ProcessEnding\(\)](#) panggilan sebelum mematikan proses server.

Tipe: `void OnProcessTerminateDelegate()`

Wajib: Ya

onHealthCheck

Nama fungsi callback yang Amazon GameLift Servers layanan memanggil untuk meminta laporan status kesehatan dari proses server. Amazon GameLift Servers memanggil fungsi ini setiap 60 detik. Setelah memanggil fungsi ini Amazon GameLift Servers menunggu 60 detik untuk respons, dan jika tidak ada yang diterima. mencatat proses server sebagai tidak sehat.

Tipe: bool OnHealthCheckDelegate()

Wajib: Ya

onUpdateGameSesi

Nama fungsi callback yang Amazon GameLift Servers layanan memanggil untuk meneruskan objek sesi game yang diperbarui ke proses server. Amazon GameLift Servers memanggil fungsi ini ketika permintaan [pengisian ulang kecocokan](#) telah diproses untuk memberikan data mak comblang yang diperbarui. Ini melewati [GameSession](#) objek, pembaruan status (updateReason), dan ID tiket isi ulang pertandingan.

Tipe: void OnUpdateGameSessionDelegate (UpdateGameSession
updateGameSession)

Wajib: Tidak

StartMatchBackfillRequest

Jenis data ini digunakan untuk mengirim permintaan backfill matchmaking. Informasi tersebut dikomunikasikan kepada Amazon GameLift Servers layanan dalam [StartMatchBackfill\(\)](#) panggilan.

Daftar Isi

GameSessionArn

Pengidentifikasi sesi game yang unik. Metode SDK [GetGameSessionId\(\)](#) mengembalikan pengenalan dalam format ARN.

Tipe: String

Diperlukan: Ya

MatchmakingConfigurationArn

Pengidentifikasi unik, dalam bentuk ARN, yang akan digunakan matchmaker untuk permintaan ini. Untuk menemukan matchmaker yang digunakan untuk membuat sesi game asli, lihat di objek sesi game, di properti data matchmaker. Pelajari selengkapnya tentang data matchmaker di [Bekerja dengan data matchmaker](#).

Tipe: String

Diperlukan: Ya

Pemain

Satu set data yang mewakili semua pemain yang saat ini dalam sesi game. Matchmaker menggunakan informasi ini untuk mencari pemain baru yang cocok untuk pemain saat ini. Lihat [Amazon GameLift Servers Panduan Referensi API](#) untuk deskripsi format objek Player. Untuk menemukan atribut pemain IDs, dan tugas tim, lihat objek sesi permainan, di properti data mak comblang. Jika latensi digunakan oleh matchmaker, kumpulkan latensi yang diperbarui untuk wilayah saat ini dan sertakan dalam data masing-masing pemain.

Jenis: [Pemain](#)[]

Wajib: Ya

TicketId

Pengidentifikasi unik untuk tiket permintaan matchmaking atau backfill match. Jika tidak ada nilai yang diberikan di sini, Amazon GameLift Servers akan menghasilkan satu dalam bentuk UUID. Gunakan pengidentifikasi ini untuk melacak status tiket backfill match atau membatalkan permintaan jika diperlukan.

Tipe: String

Wajib: Tidak

StopMatchBackfillRequest

Jenis data ini digunakan untuk membatalkan permintaan backfill matchmaking. Informasi tersebut dikomunikasikan kepada Amazon GameLift Servers layanan dalam [StopMatchBackfill\(\)](#) panggilan.

Daftar Isi

GameSessionArn

Pengidentifikasi sesi game unik yang terkait dengan permintaan yang dibatalkan.

Tipe: String

Diperlukan: Ya

MatchmakingConfigurationArn

Pengidentifikasi unik dari matchmaker sebagai tujuan pengiriman permintaan ini.

Tipe: String

Diperlukan: Ya

TicketId

Pengidentifikasi unik dari tiket backfill match yang akan dibatalkan.

Tipe: String

Diperlukan: Ya

C# server SDK untuk Amazon GameLift Servers 4.x - Jenis data

Topik

- [AcceptPlayerSession\(\)](#)
- [ActivateGameSession\(\)](#)
- [DescribePlayerSessions\(\)](#)
- [GetGameSessionId\(\)](#)
- [GetInstanceCertificate\(\)](#)
- [GetSdkVersion\(\)](#)
- [GetTerminationTime\(\)](#)
- [InitSDK\(\)](#)
- [ProcessEnding\(\)](#)
- [ProcessReady\(\)](#)
- [RemovePlayerSession\(\)](#)
- [StartMatchBackfill\(\)](#)
- [StopMatchBackfill\(\)](#)
- [TerminateGameSession\(\)](#)
- [UpdatePlayerSessionCreationPolicy\(\)](#)

AcceptPlayerSession()

Memberitahu Amazon GameLift Servers layanan bahwa pemain dengan ID sesi pemain tertentu telah terhubung ke proses server dan perlu validasi. Amazon GameLift Serversmemverifikasi bahwa

ID sesi pemain valid—yaitu, ID pemain telah memesan slot pemain dalam sesi permainan. Setelah divalidasi, Amazon GameLift Servers ubah status slot pemain dari RESERVED ke ACTIVE.

Sintaks

```
GenericOutcome AcceptPlayerSession(String playerId)
```

Parameter

playerSessionId

ID unik yang dikeluarkan oleh Amazon GameLift Servers saat sesi pemain baru dibuat. ID sesi pemain ditentukan dalam `PlayerSession` objek, yang dikembalikan sebagai respons terhadap panggilan klien ke tindakan GameLift API [StartGameSessionPlacement](#), [CreateGameSession](#), [DescribeGameSessionPlacement](#), atau [DescribePlayerSessions](#).

Tipe: String

Diperlukan: Ya

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Contoh ini menggambarkan fungsi untuk menangani permintaan koneksi, termasuk memvalidasi dan menolak sesi pemain yang tidak valid. 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()

Memberitahu Amazon GameLift Servers layanan bahwa proses server telah mengaktifkan sesi permainan dan sekarang siap menerima koneksi pemain. Tindakan ini harus dipanggil sebagai bagian dari fungsi callback `onStartGameSession()`, setelah semua inisialisasi sesi game selesai.

Sintaks

```
GenericOutcome ActivateGameSession()
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Contoh ini menunjukkan `ActivateGameSession()` yang dipanggil sebagai bagian dari fungsi delegasi `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()

Mengambil data sesi pemain, termasuk pengaturan, metadata sesi, dan data pemain. Gunakan tindakan ini untuk mendapatkan informasi untuk satu sesi pemain, untuk semua sesi pemain dalam sesi game, atau untuk semua sesi pemain yang terkait dengan ID pemain tunggal.

Sintaks

```
DescribePlayerSessionsOutcome DescribePlayerSessions(DescribePlayerSessionsRequest  
describePlayerSessionsRequest)
```

Parameter

describePlayerSessionsPermintaan

Sebuah objek [DescribePlayerSessionsRequest](#) yang menjelaskan sesi pemain mana yang diambil.

Wajib: Ya

Nilai yang dikembalikan

Jika berhasil, ini mengembalikan objek `DescribePlayerSessionsOutcome` yang berisi satu set objek sesi pemain yang sesuai dengan parameter permintaan. Objek sesi pemain memiliki struktur yang identik dengan tipe [PlayerSession](#) data AWS SDK Amazon GameLift Servers API.

Contoh

Contoh ini menggambarkan permintaan untuk semua sesi pemain yang secara aktif terhubung ke sesi game tertentu. Dengan menghilangkan `NextToken` dan menyetel nilai `Batas` ke 10, Amazon GameLift Servers akan mengembalikan catatan sesi 10 pemain pertama yang cocok dengan permintaan.

```
// 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()

Mengambil ID dari sesi game yang saat ini sedang di-host oleh proses server, jika proses server aktif.

Untuk proses idle yang belum diaktifkan dengan sesi permainan, panggilan mengembalikan `Success = True` dan `GameSessionId = ""` (string kosong).

Sintaks

```
AwsStringOutcome GetGameSessionId()
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai yang dikembalikan

Jika berhasil, ini mengembalikan ID sesi game sebagai objek `AwsStringOutcome`. Jika tidak berhasil, ini mengembalikan pesan kesalahan.

Contoh

```
var getGameSessionIdOutcome = GameLiftServerAPI.GetGameSessionId();
```

GetInstanceCertificate()

Mengambil lokasi file sertifikat TLS yang dikodekan pem-encode yang terkait dengan armada dan instance-nya. AWS Certificate Manager menghasilkan sertifikat ini saat Anda membuat armada baru dengan konfigurasi sertifikat yang disetel ke `GENERATED`. Gunakan sertifikat ini untuk membuat koneksi yang aman dengan client game dan untuk mengenkripsi komunikasi client/server.

Sintaks

```
GetInstanceCertificateOutcome GetInstanceCertificate();
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai yang dikembalikan

Jika berhasil, mengembalikan `GetInstanceCertificateOutcome` objek yang berisi lokasi file sertifikat TLS armada dan rantai sertifikat, yang disimpan pada instance. File sertifikat root, diekstrak dari rantai sertifikat, juga disimpan pada instance. Jika tidak berhasil, ini mengembalikan pesan kesalahan.

Untuk informasi selengkapnya tentang sertifikat dan data rantai sertifikat, lihat [Elemen GetCertificate Respons](#) di Referensi AWS Certificate Manager API.

Contoh

```
var getInstanceCertificateOutcome = GameLiftServerAPI.GetInstanceCertificate();
```

GetSdkVersion()

Mengembalikan nomor versi SDK saat ini yang dibangun ke dalam proses server.

Sintaks

```
AwsStringOutcome GetSdkVersion()
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai yang dikembalikan

Jika berhasil, ini mengembalikan versi SDK saat ini sebagai objek `AwsStringOutcome`. String yang dikembalikan mencakup nomor versi saja (mis. "3.1.5"). Jika tidak berhasil, ini mengembalikan pesan kesalahan.

Contoh

```
var getSdkVersionOutcome = GameLiftServerAPI.GetSdkVersion();
```

GetTerminationTime()

Mengembalikan waktu yang merupakan jadwal proses server akan ditutup, jika waktu penghentian tersedia. Proses server mengambil tindakan ini setelah menerima `onProcessTerminate()` panggilan balik dari Amazon GameLift Servers layanan. Amazon GameLift Servers [dapat memanggil](#)

[onProcessTerminate\(\)](#) untuk alasan berikut: (1) untuk kesehatan yang buruk (proses server telah melaporkan kesehatan port atau belum merespons Amazon GameLift Servers, (2) saat menghentikan instance selama acara scale-down, atau (3) ketika sebuah instance sedang dihentikan karena gangguan spot-instance.

Jika proses telah menerima callback `onProcessTerminate()`, nilai yang dikembalikan adalah perkiraan waktu penghentian. Jika proses belum menerima callback `onProcessTerminate()`, pesan kesalahan dikembalikan. Pelajari selengkapnya tentang [mematikan proses server](#).

Sintaks

```
AwsDateTimeOutcome GetTerminationTime()
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai yang dikembalikan

Jika berhasil, ini mengembalikan waktu penghentian sebagai objek `AwsDateTimeOutcome`. Nilainya adalah waktu penghentian, dinyatakan dalam detik yang telah berlalu sejak 0001 00:00:00. Misalnya, nilai waktu tanggal 2020-09-13 12:26:40 -000Z sama dengan 6373559680000000000 detik. Jika tidak ada waktu penghentian tersedia, pesan kesalahan ditampilkan.

Contoh

```
var getTerminationTimeOutcome = GameLiftServerAPI.GetTerminationTime();
```

InitSDK()

Menginisialisasi Amazon GameLift Servers SDK. Metode ini harus dipanggil saat peluncuran, sebelum inisialisasi Amazon GameLift Servers terkait lainnya terjadi.

Sintaks

```
InitSDKOutcome InitSDK()
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai yang dikembalikan

Jika berhasil, mengembalikan `InitSdkOutcome` objek yang menunjukkan bahwa proses server siap untuk memanggil [ProcessReady\(\)](#).

Contoh

```
var initSDKOutcome = GameLiftServerAPI.InitSDK();
```

ProcessEnding()

Memberitahu Amazon GameLift Servers layanan bahwa proses server dimatikan. Metode ini harus dipanggil setelah semua tugas pembersihan lainnya, termasuk mematikan semua sesi game aktif. Metode ini harus keluar dengan kode keluar 0; sebuah kode non-nol menghasilkan pesan kejadian bahwa proses tidak keluar dengan bersih.

Setelah metode keluar dengan kode 0, Anda dapat mengakhiri proses dengan kode keluar yang sukses. Anda juga dapat keluar dari proses dengan kode kesalahan. Jika Anda keluar dengan kode kesalahan, kejadian armada akan menunjukkan proses yang dihentikan secara tidak normal (`SERVER_PROCESS_TERMINATED_UNHEALTHY`).

Sintaks

```
GenericOutcome ProcessEnding()
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

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

```
Environment.Exit(errorCode);
```

ProcessReady()

Memberitahu Amazon GameLift Servers layanan bahwa proses server siap untuk meng-host sesi permainan. Panggil metode ini setelah berhasil meminta [InitSDK\(\)](#) dan menyelesaikan tugas penyiapan yang diperlukan sebelum proses server dapat menjadi host sesi game. Metode ini harus dipanggil hanya satu kali per proses.

Sintaks

```
GenericOutcome ProcessReady(ProcessParameters processParameters)
```

Parameter

processParameters

Sebuah objek [ProcessParameters](#) yang mengomunikasikan informasi berikut tentang proses server:

- Nama metode callback, diimplementasikan dalam kode server game, yang dipanggil Amazon GameLift Servers layanan untuk berkomunikasi dengan proses server.
- Nomor port yang didengarkan oleh proses server.
- Jalur ke file khusus sesi game apa pun yang Amazon GameLift Servers ingin Anda tangkap dan simpan.

Wajib: Ya

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Contoh ini menggambarkan panggilan [ProcessReady\(\)](#) dan mendelegasi implementasi fungsi.

```
// 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
        written by the game server
        {
            "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()

Memberitahu Amazon GameLift Servers layanan bahwa pemain dengan ID sesi pemain tertentu telah terputus dari proses server. Sebagai tanggapan, Amazon GameLift Servers mengubah slot pemain menjadi tersedia, yang memungkinkannya ditugaskan ke pemain baru.

Sintaks

```
GenericOutcome RemovePlayerSession(String playerId)
```

Parameter

playerSessionId

ID unik yang dikeluarkan oleh Amazon GameLift Servers saat sesi pemain baru dibuat. ID sesi pemain ditentukan dalam `PlayerSession` objek, yang dikembalikan sebagai respons terhadap panggilan klien ke tindakan GameLift API [StartGameSessionPlacement](#), [CreateGameSession](#), [DescribeGameSessionPlacement](#), atau [DescribePlayerSessions](#).

Tipe: String

Diperlukan: Ya

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

```
Aws::GameLift::GenericOutcome disconnectOutcome =  
    Aws::GameLift::Server::RemovePlayerSession(playerSessionId);
```

StartMatchBackfill()

Mengirim permintaan untuk menemukan pemain baru untuk slot terbuka dalam sesi permainan yang dibuat dengan `FlexMatch`. Lihat juga aksi AWS SDK [StartMatchBackfill\(\)](#). Dengan tindakan ini, permintaan backfill match dapat dimulai dengan proses server game yang menjadi host sesi game. Pelajari lebih lanjut tentang [fitur FlexMatch isi ulang](#).

Tindakan ini asinkron. Jika pemain baru berhasil dicocokkan, Amazon GameLift Servers layanan akan mengirimkan data mak comblang yang diperbarui menggunakan fungsi panggilan balik.

OnUpdateGameSession()

Proses server hanya dapat melakukan satu permintaan backfill match yang aktif dalam satu waktu. Untuk mengirim permintaan baru, panggil [StopMatchBackfill\(\)](#) terlebih dahulu untuk membatalkan permintaan asli.

Sintaks

```
StartMatchBackfillOutcome StartMatchBackfill (StartMatchBackfillRequest  
startBackfillRequest);
```

Parameter

StartMatchBackfillRequest

Sebuah objek [StartMatchBackfillRequest](#) yang mengkomunikasikan informasi berikut:

- ID tiket untuk ditetapkan ke permintaan backfill. Informasi ini opsional; jika tidak ada ID yang diberikan, Amazon GameLift Servers akan dibuat secara otomatis.
- Matchmaker untuk dikirim permintaan. ARN konfigurasi penuh diperlukan. Nilai ini dapat diperoleh dari data matchmaker sesi game.
- ID dari sesi game yang sedang di-backfill.
- Data matchmaking yang tersedia untuk pemain sesi game saat ini.

Wajib: Ya

Nilai yang dikembalikan

Mengembalikan StartMatchBackfillOutcome objek dengan ID tiket isi ulang pertandingan atau kegagalan dengan pesan kesalahan.

Contoh

```
// 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()

Membatalkan permintaan backfill match aktif yang dibuat dengan [StartMatchBackfill\(\)](#). Lihat juga aksi AWS SDK [StopMatchmaking\(\)](#). Pelajari lebih lanjut tentang [fitur FlexMatch isi ulang](#).

Sintaks

```
GenericOutcome StopMatchBackfill (StopMatchBackfillRequest stopBackfillRequest);
```

Parameter

StopMatchBackfillRequest

Sebuah objek [StopMatchBackfillRequest](#) yang mengidentifikasi tiket matchmaking untuk membatalkan:

- ID tiket yang ditetapkan ke permintaan backfill yang dibatalkan
- matchmaker yang dikirim permintaan backfill
- sesi game yang terkait dengan permintaan backfill

Wajib: Ya

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

```
// 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()

Metode ini tidak lagi digunakan dengan versi 4.0.1. Sebaliknya, proses server harus memanggil [ProcessEnding\(\)](#) setelah sesi permainan berakhir.

Memberitahu Amazon GameLift Servers layanan bahwa proses server telah mengakhiri sesi permainan saat ini. Tindakan ini dipanggil ketika proses server akan tetap aktif dan siap untuk menjadi host sesi game baru. Ini harus dipanggil hanya setelah prosedur penghentian sesi permainan Anda selesai, karena ini menandakan Amazon GameLift Servers bahwa proses server segera tersedia untuk meng-host sesi permainan baru.

Tindakan ini tidak dipanggil jika proses server akan dimatikan setelah sesi game berhenti. Sebagai gantinya, panggil [ProcessEnding\(\)](#) untuk memberi tahu bahwa sesi game dan proses server akan berakhir.

Sintaks

```
GenericOutcome TerminateGameSession()
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Contoh ini menggambarkan proses server pada akhir sesi game.

```
// 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()

Memperbarui kemampuan sesi game saat ini untuk menerima sesi pemain baru. Sesi game dapat diatur untuk menerima atau menolak semua sesi pemain baru. (Lihat juga tindakan [UpdateGameSession\(\)](#) di Referensi API Amazon GameLift Servers Layanan).

Sintaks

```
GenericOutcome UpdatePlayerSessionCreationPolicy(PlayerSessionCreationPolicy  
    playerSessionPolicy)
```

Parameter

newPlayerSessionKebijakan

Nilai string yang menunjukkan apakah sesi game menerima pemain baru.

Jenis: Enum [PlayerSessionCreationPolicy](#). Nilai yang valid meliputi:

- ACCEPT_ALL — Menerima semua sesi pemain baru.
- DENY_ALL — Menolak semua sesi pemain baru.

Wajib: Ya

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Contoh ini menetapkan kebijakan bergabung sesi game saat ini untuk menerima semua pemain.

```
var updatePlayerSessionCreationPolicyOutcome =  
  
    GameLiftServerAPI.UpdatePlayerSessionCreationPolicy(PlayerSessionCreationPolicy.ACCEPT_ALL);
```

Server SDK (Unreal) untuk Amazon GameLift Servers -- Tindakan

Gunakan SDK server untuk Unreal untuk mengintegrasikan game multipemain Anda untuk hosting Amazon GameLift Servers. Untuk panduan tentang proses integrasi, lihat [Tambahkan Amazon GameLift Servers ke server game Anda](#).

Note

Referensi ini untuk versi SDK server yang lebih lama Amazon GameLift Servers. Untuk versi terbaru, lihat [Server C ++ \(Tidak nyata\) SDK 5.x untuk -- Tindakan Amazon GameLift Servers](#).

API ini didefinisikan dalam `GameLiftServerSDK.h` dan `GameLiftServerSDKModels.h`.

Untuk menyiapkan plugin Unreal Engine dan melihat contoh kode [Integrasikan Amazon GameLift Servers ke dalam proyek Unreal Engine](#).

Server SDK (Unreal) untuk Amazon GameLift Servers - Jenis data

Gunakan Amazon GameLift Servers server SDK untuk referensi Unreal untuk mengintegrasikan game multipemain Anda untuk hosting Amazon GameLift Servers. Untuk panduan tentang proses integrasi, lihat [Tambahkan Amazon GameLift Servers ke server game Anda](#).

Note

Referensi ini untuk versi SDK server yang lebih lama Amazon GameLift Servers. Untuk versi terbaru, lihat [Server C ++ \(Tidak nyata\) SDK 5.x untuk Amazon GameLift Servers -- Tipe data](#).

API ini didefinisikan dalam `GameLiftServerSDK.h` dan `GameLiftServerSDKModels.h`.

Untuk menyiapkan plugin Unreal Engine dan melihat contoh kode [Integrasikan Amazon GameLift Servers ke dalam proyek Unreal Engine](#).

[Server SDK \(Unreal\) untuk Amazon GameLift Servers -- Tindakan](#)

Topik

- [FDescribePlayerSessionsRequest](#)
- [FProcessParameter](#)
- [FStartMatchBackfillRequest](#)
- [FStopMatchBackfillRequest](#)

FDescribePlayerSessionsRequest

Jenis data ini digunakan untuk menentukan sesi pemain untuk diambil. Anda bisa menggunakannya sebagai berikut:

- Menyediakan `PlayerSessionId` untuk meminta sesi pemain tertentu.
- Berikan permintaan `GameSessionId` untuk semua sesi pemain dalam sesi permainan yang ditentukan.
- Berikan permintaan `PlayerId` untuk semua sesi pemain untuk pemain yang ditentukan.

Untuk koleksi sesi pemain yang besar, gunakan parameter pemberian nomor halaman untuk mengambil hasil dalam blok berurutan.

Daftar Isi

GameSessionId

Pengidentifikasi sesi game yang unik. Gunakan parameter ini untuk meminta semua sesi pemain untuk sesi game yang ditentukan. Format ID sesi game adalah sebagai berikut:
`arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string>`. Nilai `<ID string>` adalah string ID kustom atau (jika salah satu ditentukan saat sesi game dibuat) string yang dihasilkan.

Tipe: String

Wajib: Tidak

Kuota

Jumlah hasil maksimum yang akan dikembalikan. Gunakan parameter ini dengan `NextToken` untuk mendapatkan hasil sebagai satu set halaman berurutan. Jika ID sesi pemain ditentukan, parameter ini diabaikan.

Jenis: Integer

Wajib: Tidak

NextToken

Token yang menunjukkan awal dari halaman berurutan berikutnya dari hasil. Gunakan token yang dikembalikan dengan panggilan sebelumnya untuk tindakan ini. Untuk menentukan awal set hasil, jangan menentukan nilai. Jika ID sesi pemain ditentukan, parameter ini diabaikan.

Tipe: String

Wajib: Tidak

PlayerId

Pengidentifikasi unik untuk pemain. Pemain IDs ditentukan oleh pengembang. Lihat [Hasilkan pemain IDs](#).

Tipe: String

Wajib: Tidak

PlayerSessionId

Pengidentifikasi unik untuk sesi pemain.

Tipe: String

Wajib: Tidak

PlayerSessionStatusFilter

Status sesi pemain untuk mem-filter hasil. Kemungkinan status sesi pemain meliputi:

- RESERVED — Permintaan sesi pemain telah diterima, namun pemain belum terhubung ke proses server dan/atau divalidasi.
- ACTIVE — Pemain telah divalidasi oleh proses server dan saat ini terhubung.
- COMPLETED — Sambungan pemain telah terputus.
- TIMEDOUT — Permintaan sesi pemain diterima, namun pemain tidak terhubung dan/atau tidak divalidasi dalam batas waktu (60 detik).

Tipe: String

Wajib: Tidak

FProcessParameter

Tipe data ini berisi kumpulan parameter yang dikirim ke Amazon GameLift Servers layanan dalam [ProcessReady\(\)](#) panggilan.

Daftar Isi

port

Nomor port yang akan didengarkan proses server untuk koneksi pemain baru. Nilai harus masuk ke dalam kisaran port yang dikonfigurasi untuk setiap armada yang men-deploy build server game ini. Nomor port ini termasuk dalam sesi game dan objek sesi pemain, yang digunakan sesi game saat menghubungkan ke proses server.

Jenis: Integer

Wajib: Ya

logParameters

Obyek dengan daftar jalur direktori untuk file log sesi game.

Jenis: TArray < FString >

Wajib: Tidak

onStartGameSesi

Nama fungsi callback yang Amazon GameLift Servers layanan memanggil untuk mengaktifkan sesi permainan baru. Amazon GameLift Servers memanggil fungsi ini sebagai tanggapan atas permintaan klien [CreateGameSession](#). Fungsi callback mengambil [GameSession](#) objek (didefinisikan dalam Amazon GameLift Servers Referensi API Layanan).

Jenis: FOn StartGameSession

Wajib: Ya

onProcessTerminate

Nama fungsi callback yang Amazon GameLift Servers layanan memanggil untuk memaksa proses server dimatikan. Setelah memanggil fungsi ini, Amazon GameLift Servers menunggu lima menit

hingga proses server dimatikan dan merespons dengan [ProcessEnding\(\)](#) panggilan sebelum mematikan proses server.

Jenis: FSimple Delegasi

Wajib: Tidak

onHealthCheck

Nama fungsi callback yang Amazon GameLift Servers layanan memanggil untuk meminta laporan status kesehatan dari proses server. Amazon GameLift Servers memanggil fungsi ini setiap 60 detik. Setelah memanggil fungsi ini Amazon GameLift Servers menunggu 60 detik untuk respons, dan jika tidak ada yang diterima, mencatat proses server sebagai tidak sehat.

Jenis: FOn HealthCheck

Wajib: Tidak

onUpdateGameSesi

Nama fungsi callback yang Amazon GameLift Servers layanan memanggil untuk meneruskan objek sesi game yang diperbarui ke proses server. Amazon GameLift Servers memanggil fungsi ini ketika permintaan [pengisian ulang kecocokan](#) telah diproses untuk memberikan data mak comblang yang diperbarui. Ini melewati [GameSession](#) objek, pembaruan status (updateReason), dan ID tiket isi ulang pertandingan.

Jenis: FOn UpdateGameSession

Wajib: Tidak

FStartMatchBackfillRequest

Jenis data ini digunakan untuk mengirim permintaan backfill matchmaking. Informasi tersebut dikomunikasikan kepada Amazon GameLift Servers layanan dalam [StartMatchBackfill\(\)](#) panggilan.

Daftar Isi

GameSessionArn

Pengidentifikasi sesi game yang unik. Tindakan API [GetGameSessionId\(\)](#) mengembalikan pengidentifikasi dalam format ARN.

Jenis: FString

Wajib: Ya

MatchmakingConfigurationArn

Pengidentifikasi unik, dalam bentuk ARN, yang akan digunakan matchmaker untuk permintaan ini. Untuk menemukan matchmaker yang digunakan untuk membuat sesi game asli, lihat di objek sesi game, di properti data matchmaker. Pelajari selengkapnya tentang data matchmaker di [Bekerja dengan data matchmaker](#).

Jenis: FString

Wajib: Ya

Pemain

Satu set data yang mewakili semua pemain yang saat ini dalam sesi game. Matchmaker menggunakan informasi ini untuk mencari pemain baru yang cocok untuk pemain saat ini. Lihat Amazon GameLift Servers Panduan Referensi API untuk deskripsi format objek Player. Untuk menemukan atribut pemain IDs, dan tugas tim, lihat objek sesi permainan, di properti data mak comblang. Jika latensi digunakan oleh matchmaker, kumpulkan latensi yang diperbarui untuk wilayah saat ini dan sertakan dalam data masing-masing pemain.

Jenis: TArray [< FPlayer >](#)

Wajib: Ya

TicketId

Pengidentifikasi unik untuk tiket permintaan matchmaking atau backfill match. Jika tidak ada nilai yang diberikan di sini, Amazon GameLift Servers akan menghasilkan satu dalam bentuk UUID. Gunakan pengidentifikasi ini untuk melacak status tiket backfill match atau membatalkan permintaan jika diperlukan.

Jenis: FString

Wajib: Tidak

FStopMatchBackfillRequest

Jenis data ini digunakan untuk membatalkan permintaan backfill matchmaking. Informasi tersebut dikomunikasikan kepada Amazon GameLift Servers layanan dalam [StopMatchBackfill\(\)](#) panggilan.

Daftar Isi

GameSessionArn

Pengidentifikasi sesi game unik yang terkait dengan permintaan yang dibatalkan.

Jenis: FString

Wajib: Ya

MatchmakingConfigurationArn

Pengidentifikasi unik dari matchmaker sebagai tujuan pengiriman permintaan ini.

Jenis: FString

Wajib: Ya

TicketId

Pengidentifikasi unik dari tiket backfill match yang akan dibatalkan.

Jenis: FString

Wajib: Ya

[Server SDK \(Unreal\) untuk Amazon GameLift Servers - Jenis data](#)

Topik

- [AcceptPlayerSession\(\)](#)
- [ActivateGameSession\(\)](#)
- [DescribePlayerSessions\(\)](#)
- [GetGameSessionId\(\)](#)
- [GetInstanceCertificate\(\)](#)
- [GetSdkVersion\(\)](#)
- [InitSDK\(\)](#)
- [ProcessEnding\(\)](#)
- [ProcessReady\(\)](#)
- [RemovePlayerSession\(\)](#)
- [StartMatchBackfill\(\)](#)

- [StopMatchBackfill\(\)](#)
- [TerminateGameSession\(\)](#)
- [UpdatePlayerSessionCreationPolicy\(\)](#)

AcceptPlayerSession()

Memberitahukan Amazon GameLift Servers layanan yang pemain dengan ID sesi pemain tertentu telah terhubung ke proses server dan membutuhkan validasi. Amazon GameLift Servers memverifikasi bahwa ID sesi pemain valid—yaitu, ID pemain telah memesan slot pemain dalam sesi permainan. Setelah divalidasi, Amazon GameLift Servers mengubah status slot pemain dari RESERVED ke ACTIVE.

Sintaks

```
FGameLiftGenericOutcome AcceptPlayerSession(const FString& playerId)
```

Parameter

playerSessionId

ID unik yang dikeluarkan oleh Amazon GameLift Servers layanan dalam menanggapi panggilan ke AWS SDK Amazon GameLift Servers Tindakan API [CreatePlayerSession](#). Client game mereferensi ID ini saat menghubungkan ke proses server.

Jenis: FString

Wajib: Ya

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

ActivateGameSession()

Memberitahukan Amazon GameLift Servers layanan bahwa proses server telah mengaktifkan sesi permainan dan sekarang siap menerima koneksi pemain. Tindakan ini harus dipanggil sebagai bagian dari fungsi callback `onStartGameSession()`, setelah semua inisialisasi sesi game selesai.

Sintaks

```
FGameLiftGenericOutcome ActivateGameSession()
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

DescribePlayerSessions()

Mengambil data sesi pemain, termasuk pengaturan, metadata sesi, dan data pemain. Gunakan tindakan ini untuk mendapatkan informasi untuk satu sesi pemain, untuk semua sesi pemain dalam sesi game, atau untuk semua sesi pemain yang terkait dengan ID pemain tunggal.

Sintaks

```
FGameLiftDescribePlayerSessionsOutcome DescribePlayerSessions(const  
FGameLiftDescribePlayerSessionsRequest &describePlayerSessionsRequest)
```

Parameter

describePlayerSessionsPermintaan

Sebuah objek [FDescribePlayerSessionsRequest](#) yang menjelaskan sesi pemain mana yang diambil.

Wajib: Ya

Nilai yang dikembalikan

Jika berhasil, ini mengembalikan objek [FDescribePlayerSessionsRequest](#) yang berisi satu set objek sesi pemain yang sesuai dengan parameter permintaan. Objek sesi pemain memiliki struktur yang identik dengan AWS SDK Amazon GameLift Servers Jenis [PlayerSession](#) data API.

GetGameSessionId()

Mengambil ID dari sesi game yang saat ini sedang di-host oleh proses server, jika proses server aktif.

Sintaks

```
FGameLiftStringOutcome GetGameSessionId()
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai yang dikembalikan

Jika berhasil, ini mengembalikan ID sesi game sebagai objek `FGameLiftStringOutcome`. Jika tidak berhasil, ini mengembalikan pesan kesalahan.

GetInstanceCertificate()

Mengambil lokasi file sertifikat TLS yang dikodekan pem-encode yang terkait dengan armada dan instance-nya. AWS Certificate Manager menghasilkan sertifikat ini saat Anda membuat armada baru dengan konfigurasi sertifikat yang disetel ke `GENERATED`. Gunakan sertifikat ini untuk membuat koneksi yang aman dengan client game dan untuk mengenkripsi komunikasi client/server.

Sintaks

```
FGameLiftGetInstanceCertificateOutcome GetInstanceCertificate()
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai yang dikembalikan

Jika berhasil, mengembalikan `GetInstanceCertificateOutcome` objek yang berisi lokasi file sertifikat TLS armada dan rantai sertifikat, yang disimpan pada instance. File sertifikat root, diekstrak dari rantai sertifikat, juga disimpan pada instance. Jika tidak berhasil, ini mengembalikan pesan kesalahan.

Untuk informasi selengkapnya tentang sertifikat dan data rantai sertifikat, lihat [Elemen GetCertificate Respons](#) di Referensi AWS Certificate Manager API.

GetSdkVersion()

Mengembalikan nomor versi SDK saat ini yang dibangun ke dalam proses server.

Sintaks

```
FGameLiftStringOutcome GetSdkVersion();
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai yang dikembalikan

Jika berhasil, ini mengembalikan versi SDK saat ini sebagai objek `FGameLiftStringOutcome`. String yang dikembalikan mencakup nomor versi saja (mis. "3.1.5"). Jika tidak berhasil, ini mengembalikan pesan kesalahan.

Contoh

```
Aws::GameLift::AwsStringOutcome SdkVersionOutcome =  
    Aws::GameLift::Server::GetSdkVersion();
```

InitSDK()

Menginisialisasi Amazon GameLift Servers SDK. Metode ini harus dipanggil saat peluncuran, sebelum yang lain Amazon GameLift ServersTerjadi inisialisasi terkait.

Sintaks

```
FGameLiftGenericOutcome InitSDK()
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

ProcessEnding()

Memberitahukan Amazon GameLift Servers layanan yang proses server dimatikan. Metode ini harus dipanggil setelah semua tugas pembersihan lainnya, termasuk mematikan semua sesi game aktif.

Metode ini harus keluar dengan kode keluar 0; sebuah kode non-nol menghasilkan pesan kejadian bahwa proses tidak keluar dengan bersih.

Sintaks

```
FGameLiftGenericOutcome ProcessEnding()
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

ProcessReady()

Memberitahukan Amazon GameLift Servers layanan bahwa proses server siap untuk meng-host sesi permainan. Panggil metode ini setelah berhasil meminta [InitSDK\(\)](#) dan menyelesaikan tugas penyiapan yang diperlukan sebelum proses server dapat menjadi host sesi game. Metode ini harus dipanggil hanya satu kali per proses.

Sintaks

```
FGameLiftGenericOutcome ProcessReady(FProcessParameters &processParameters)
```

Parameter

FProcessParameter

Sebuah objek [FProcessParameter](#) yang mengomunikasikan informasi berikut tentang proses server:

- Nama metode callback, diimplementasikan dalam kode server game, bahwa Amazon GameLift Servers layanan memanggil untuk berkomunikasi dengan proses server.
- Nomor port yang didengarkan oleh proses server.
- Jalur ke file khusus sesi game apa pun yang Anda inginkan Amazon GameLift Servers untuk menangkap dan menyimpan.

Wajib: Ya

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Lihat kode sampel di [Menggunakan Plugin Unreal Engine](#).

RemovePlayerSession()

Memberitahukan Amazon GameLift Servers layanan bahwa pemain dengan ID sesi pemain yang ditentukan telah terputus dari proses server. Sebagai tanggapan, Amazon GameLift Servers mengubah slot pemain menjadi tersedia, yang memungkinkannya ditugaskan ke pemain baru.

Sintaks

```
FGameLiftGenericOutcome RemovePlayerSession(const FString& playerId)
```

Parameter

playerSessionId

ID unik yang dikeluarkan oleh Amazon GameLift Servers layanan dalam menanggapi panggilan ke AWS SDK Amazon GameLift Servers Tindakan API [CreatePlayerSession](#). Client game mereferensi ID ini saat menghubungkan ke proses server.

Jenis: FString

Wajib: Ya

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

StartMatchBackfill()

Mengirim permintaan untuk menemukan pemain baru untuk slot terbuka dalam sesi permainan yang dibuat dengan FlexMatch. Lihat juga aksi AWS SDK [StartMatchBackfill\(\)](#). Dengan tindakan ini,

permintaan backfill match dapat dimulai dengan proses server game yang menjadi host sesi game. Pelajari lebih lanjut tentang [FlexMatch fitur](#) isi ulang.

Tindakan ini asinkron. Jika pemain baru berhasil dicocokkan, Amazon GameLift Servers layanan memberikan data mak comblang yang diperbarui menggunakan fungsi callback. `OnUpdateGameSession()`

Proses server hanya dapat melakukan satu permintaan backfill match yang aktif dalam satu waktu. Untuk mengirim permintaan baru, panggil [StopMatchBackfill\(\)](#) terlebih dahulu untuk membatalkan permintaan asli.

Sintaks

```
FGameLiftStringOutcome StartMatchBackfill (FStartMatchBackfillRequest  
&startBackfillRequest);
```

Parameter

FStartMatchBackfillRequest

Sebuah objek [FStartMatchBackfillRequest](#) yang mengkomunikasikan informasi berikut:

- ID tiket untuk ditetapkan ke permintaan backfill. Informasi ini bersifat opsional; jika tidak ada ID yang diberikan, Amazon GameLift Servers akan membuat otomatis satu.
- Matchmaker untuk dikirim permintaan. ARN konfigurasi penuh diperlukan. Nilai ini dapat diperoleh dari data matchmaker sesi game.
- ID dari sesi game yang sedang di-backfill.
- Data matchmaking yang tersedia untuk pemain sesi game saat ini.

Wajib: Ya

Nilai yang dikembalikan

Jika berhasil, mengembalikan tiket backfill match sebagai objek `FGameLiftStringOutcome`. Jika tidak berhasil, ini mengembalikan pesan kesalahan. Status tiket dapat dilacak menggunakan tindakan AWS SDK [DescribeMatchmaking\(\)](#).

StopMatchBackfill()

Membatalkan permintaan backfill match aktif yang dibuat dengan [StartMatchBackfill\(\)](#). Lihat juga aksi AWS SDK [StopMatchmaking\(\)](#). Pelajari lebih lanjut tentang [FlexMatch fitur](#) isi ulang.

Sintaks

```
FGameLiftGenericOutcome StopMatchBackfill (FStopMatchBackfillRequest  
&stopBackfillRequest);
```

Parameter

StopMatchBackfillRequest

Sebuah objek [FStopMatchBackfillRequest](#) yang mengidentifikasi tiket matchmaking untuk membatalkan:

- ID tiket yang ditetapkan ke permintaan backfill yang dibatalkan
- matchmaker yang dikirim permintaan backfill
- sesi game yang terkait dengan permintaan backfill

Wajib: Ya

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

TerminateGameSession()

Metode ini tidak lagi digunakan dengan versi 4.0.1. Sebaliknya, proses server harus memanggil [ProcessEnding\(\)](#) setelah sesi permainan berakhir.

Memberitahukan Amazon GameLift Servers layanan bahwa proses server telah mengakhiri sesi permainan saat ini. Tindakan ini dipanggil ketika proses server akan tetap aktif dan siap untuk menjadi host sesi game baru. Ini harus dipanggil hanya setelah prosedur penghentian sesi permainan Anda selesai, karena itu memberi sinyal ke Amazon GameLift Servers bahwa proses server segera tersedia untuk meng-host sesi permainan baru.

Tindakan ini tidak dipanggil jika proses server akan dimatikan setelah sesi game berhenti. Sebagai gantinya, panggil [ProcessEnding\(\)](#) untuk memberi tahu bahwa sesi game dan proses server akan berakhir.

Sintaks

```
FGameLiftGenericOutcome TerminateGameSession()
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

UpdatePlayerSessionCreationPolicy()

Memperbarui kemampuan sesi game saat ini untuk menerima sesi pemain baru. Sesi game dapat diatur untuk menerima atau menolak semua sesi pemain baru. (Lihat juga [UpdateGameSession\(\)](#) aksi di Amazon GameLift Servers Referensi API Layanan).

Sintaks

```
FGameLiftGenericOutcome UpdatePlayerSessionCreationPolicy(EPlayerSessionCreationPolicy policy)
```

Parameter

Kebijakan

Nilai yang menunjukkan apakah sesi game menerima pemain baru.

Jenis: Enum `EPlayerSessionCreationPolicy`. Nilai yang valid meliputi:

- `ACCEPT_ALL` — Menerima semua sesi pemain baru.
- `DENY_ALL` — Menolak semua sesi pemain baru.

Wajib: Ya

Nilai yang dikembalikan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Amazon GameLift Servers

Amazon GameLift ServersRealtimeReferensi

Bagian ini berisi dokumentasi referensi untuk Amazon GameLift Servers Amazon GameLift Servers Realtime SDK. Ini termasuk API Realtime Klien serta panduan untuk mengonfigurasi Amazon GameLift Servers Realtime skrip Anda.

Topik

- [Amazon GameLift ServersRealtimeReferensi API klien \(C #\)](#)
- [Referensi skrip untuk Amazon GameLift ServersRealtime](#)

Amazon GameLift ServersRealtimeReferensi API klien (C #)

Gunakan API Realtime Klien untuk mempersiapkan klien game multipemain Anda untuk digunakan. Amazon GameLift Servers Amazon GameLift Servers Realtime API Klien berisi serangkaian panggilan API sinkron dan panggilan balik asinkron yang memungkinkan klien game terhubung ke Realtime server dan bertukar pesan dan data dengan klien game lain melalui server.

API ini ditentukan di pustaka berikut:

Client.cs

- [Tindakan Sinkron](#)
- [Panggilan Balik Asinkron](#)
- [Tipe Data](#)

Untuk menyiapkan API Realtime klien

1. Unduh [SDK Amazon GameLift ServersRealtime klien](#).
2. Bangun pustaka C# SDK. Cari file solusi `GameLiftRealtimeClientSdkNet45.sln`. Lihat file `README.md` untuk SDK C# Server untuk persyaratan minimum dan tambahan opsi build. Dalam IDE, muat file solusi. Untuk menghasilkan pustaka SDK, pulihkan NuGet paket dan buat solusinya.
3. Tambahkan pustaka Realtime Klien ke proyek klien game Anda.

Amazon GameLift ServersRealtimereferensi API klien (C #): Tindakan

Referensi C# Realtime Client API ini dapat membantu Anda mempersiapkan game multipemain Anda untuk digunakan dengan Amazon GameLift Servers Realtime dikerahkan pada armada. Amazon GameLift Servers

- Tindakan Sinkron
- [Panggilan Balik Asinkron](#)
- [Tipe Data](#)

Client()

Menginisialisasi client baru untuk berkomunikasi dengan server Realtime dan mengidentifikasi jenis koneksi yang akan digunakan.

Sintaks

```
public Client(ClientConfiguration configuration)
```

Parameter

clientConfiguration

Detail konfigurasi yang menentukan jenis client/server koneksi. Anda dapat memilih untuk memanggil Client() tanpa parameter ini; namun, pendekatan ini secara default menghasilkan koneksi yang tidak aman.

Tipe: [ClientConfiguration](#)

Wajib: Tidak

Nilai yang dikembalikan

Mengembalikan sebuah instans dari client Realtime untuk digunakan dengan berkomunikasi dengan server Realtime.

Connect()

Meminta koneksi ke proses server yang meng-host sesi game.

Sintaks

```
public ConnectionStatus Connect(string endpoint, int remoteTcpPort, int listenPort,
    ConnectionToken token)
```

Parameter

titik akhir

Nama DNS atau alamat IP sesi game yang akan disambungkan. Titik akhir ditentukan dalam `GameSession` objek, yang dikembalikan sebagai respons terhadap panggilan klien ke tindakan AWS SDK Amazon GameLift Servers API [StartGameSessionPlacementCreateGameSession](#), atau [DescribeGameSessions](#)

Note

Jika server Realtime berjalan pada armada dengan sertifikat TLS, Anda harus menggunakan nama DNS.

Jenis: String

Diperlukan: Ya

remoteTcpPort

Nomor port untuk koneksi TCP yang ditetapkan ke sesi game. Informasi ini ditentukan dalam `GameSession` objek, yang dikembalikan sebagai tanggapan terhadap [StartGameSessionPlacementCreateGameSession](#), atau [DescribeGameSession](#) permintaan.

Jenis: Integer

Nilai Valid: 1900 hingga 2000.

Wajib: Ya

listenPort

Nomor port yang didengarkan client game pada pesan yang dikirim menggunakan saluran UDP.

Jenis: Integer

Nilai Valid: 33400 hingga 33500.

Wajib: Ya

token

Informasi opsional yang mengidentifikasi permintaan client game kepada proses server.

Tipe: [ConnectionToken](#)

Wajib: Ya

Nilai yang dikembalikan

Mengembalikan nilai [ConnectionStatus](#) enum yang menunjukkan status koneksi klien.

Disconnect()

Memutuskan koneksi client game dari sesi game saat terhubung ke sesi game.

Sintaks

```
public void Disconnect()
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai yang dikembalikan

Metode ini tidak mengembalikan apa pun.

NewMessage()

Membuat objek pesan baru dengan kode operasi tertentu. Setelah objek pesan dikembalikan, selesaikan isi pesan dengan menentukan target, memperbarui metode pengiriman, dan menambahkan muatan data sesuai kebutuhan. Setelah selesai, kirim pesan menggunakan `SendMessage()`.

Sintaks

```
public RTMessage NewMessage(int opCode)
```

Parameter

opCode

Kode operasi yang didefinisikan developer yang mengidentifikasi kejadian atau tindakan game, seperti gerakan pemain atau notifikasi server.

Jenis: Integer

Wajib: Ya

Nilai yang dikembalikan

Mengembalikan sebuah objek [RTMessage](#) yang berisi kode operasi tertentu dan metode pengiriman default. Secara default, parameter pengiriman yang diinginkan diatur ke FAST.

SendMessage()

Mengirim pesan ke pemain atau grup menggunakan metode pengiriman yang ditentukan.

Sintaks

```
public void SendMessage(RTMessage message)
```

Parameter

pesan

Objek pesan yang menentukan target penerima, metode pengiriman, dan isi pesan.

Tipe: [RTMessage](#)

Wajib: Ya

Nilai yang dikembalikan

Metode ini tidak mengembalikan apa pun.

JoinGroup()

Menambahkan pemain ke keanggotaan grup tertentu. Grup dapat berisi salah satu pemain yang terhubung ke game. Setelah bergabung, pemain menerima semua pesan berikutnya yang dikirim ke grup dan dapat mengirim pesan ke seluruh grup.

Sintaks

```
public void JoinGroup(int targetGroup)
```

Parameter

targetGroup

ID unik yang mengidentifikasi grup sebagai cara menambahkan pemain ke dalamnya. Grup IDs didefinisikan oleh pengembang.

Jenis: Integer

Wajib: Ya

Nilai yang dikembalikan

Metode ini tidak mengembalikan apa pun. Karena permintaan ini dikirim menggunakan metode pengiriman yang andal (TCP), permintaan yang gagal akan memicu callback [OnError\(\)](#).

LeaveGroup()

Menghapus pemain dari keanggotaan grup tertentu. Setelah tidak lagi berada dalam grup, pemain tidak menerima pesan yang dikirim ke grup dan tidak dapat mengirim pesan ke seluruh grup.

Sintaks

```
public void LeaveGroup(int targetGroup)
```

Parameter

targetGroup

ID unik mengidentifikasi grup untuk menghapus pemain dari dalamnya. Grup IDs didefinisikan oleh pengembang.

Jenis: Integer

Wajib: Ya

Nilai yang dikembalikan

Metode ini tidak mengembalikan apa pun. Karena permintaan ini dikirim menggunakan metode pengiriman yang andal (TCP), permintaan yang gagal akan memicu callback [OnError\(\)](#).

RequestGroupMembership()

Meminta daftar pemain dalam grup tertentu dikirim ke client game. Setiap pemain dapat meminta informasi ini, terlepas dari apakah mereka adalah anggota grup atau tidak. Menanggapi permintaan ini, daftar keanggotaan dikirim ke client melalui callback [OnGroupMembershipUpdated\(\)](#).

Sintaks

```
public void RequestGroupMembership(int targetGroup)
```

Parameter

targetGroup

ID unik yang mengidentifikasi grup untuk mendapatkan informasi keanggotaan. Grup IDs didefinisikan oleh pengembang.

Jenis: Integer

Wajib: Ya

Nilai yang dikembalikan

Metode ini tidak mengembalikan apa pun.

Amazon GameLift ServersRealtimereferensi API klien (C #): Panggilan balik asinkron

Gunakan referensi C# Realtime Client API ini untuk membantu Anda mempersiapkan game multipemain Anda untuk digunakan dengan Amazon GameLift Servers Realtime diarahkan pada armada. Amazon GameLift Servers

- [Tindakan Sinkron](#)
- [Callback Asinkron](#)
- [Tipe Data](#)

Sebuah client game perlu menerapkan metode callback ini untuk menanggapi kejadian. RealtimeServer memanggil callback ini untuk mengirim informasi terkait game ke klien game. Callback untuk acara yang sama juga dapat diimplementasikan dengan logika permainan khusus di skrip Realtime server. Lihat [Callback skrip untuk Amazon GameLift ServersRealtime](#).

Metode callback didefinisikan dalam `ClientEvents.cs`.

OnOpen()

Dipanggil ketika proses server menerima permintaan koneksi client game dan membuka koneksi.

Sintaks

```
public void OnOpen()
```

Parameter

Metode ini tidak menggunakan parameter.

Nilai yang dikembalikan

Metode ini tidak mengembalikan apa pun.

OnClose()

Dipanggil ketika proses server mengakhiri koneksi dengan client game, seperti setelah sesi game berakhir.

Sintaks

```
public void OnClose()
```

Parameter

Metode ini tidak menggunakan parameter.

Nilai yang dikembalikan

Metode ini tidak mengembalikan apa pun.

OnError()

Dipanggil ketika terjadi kegagalan untuk permintaan API Realtime Client API. Callback ini dapat disesuaikan untuk menangani berbagai kesalahan koneksi.

Sintaks

```
private void OnError(byte[] args)
```

Parameter

Metode ini tidak menggunakan parameter.

Nilai yang dikembalikan

Metode ini tidak mengembalikan apa pun.

OnDataReceived()

Dipanggil saat klien game menerima pesan dari Realtime server. Ini adalah metode utama di mana pesan dan notifikasi diterima oleh client game.

Sintaks

```
public void OnDataReceived(DataReceivedEventArgs dataReceivedEventArgs)
```

Parameter

`dataReceivedEventArgs`

Informasi yang berkaitan dengan aktivitas pesan.

Tipe: [DataReceivedEventArgs](#)

Wajib: Ya

Nilai yang dikembalikan

Metode ini tidak mengembalikan apa pun.

OnGroupMembershipUpdated()

Dipanggil ketika keanggotaan untuk grup yang ditempati pemain telah diperbarui. Callback ini juga dipanggil ketika client memanggil `RequestGroupMembership`.

Sintaks

```
public void OnGroupMembershipUpdated(GroupMembershipEventArgs groupMembershipEventArgs)
```

Parameter

groupMembershipEventArgs

Informasi yang berkaitan dengan kegiatan keanggotaan grup.

Tipe: [GroupMembershipEventArgs](#)

Wajib: Ya

Nilai yang dikembalikan

Metode ini tidak mengembalikan apa pun.

Amazon GameLift ServersRealtimereferensi API klien (C #): Tipe data

Referensi C# Realtime Client API ini dapat membantu Anda mempersiapkan game multipemain Anda untuk digunakan dengan Amazon GameLift Servers Realtime dikerahkan pada armada. Amazon GameLift Servers

- [Tindakan Sinkron](#)
- [Panggilan Balik Asinkron](#)
- Jenis Data

ClientConfiguration

Informasi tentang bagaimana klien game terhubung ke Realtime server.

Daftar Isi

ConnectionType

Jenis client/server koneksi yang akan digunakan, baik aman atau tidak aman. Jika Anda tidak menentukan jenis koneksi, pengaturan default-nya adalah koneksi tidak aman.

Note

Saat menghubungkan ke server Realtime pada armada aman dengan sertifikat TLS, Anda harus menggunakan nilai RT_OVER_WSS_DTLS_. TLS12

Jenis: nilai `ConnectionType` [enum](#) A.

Wajib: Tidak

ConnectionToken

Informasi tentang and/or pemain klien game yang meminta koneksi dengan Realtime server.

Daftar Isi

playerSessionId

ID unik yang dikeluarkan oleh Amazon GameLift Servers saat sesi pemain baru dibuat. ID sesi pemain ditentukan dalam `PlayerSession` objek, yang dikembalikan sebagai respons terhadap panggilan klien ke tindakan GameLift API [StartGameSessionPlacement](#), [CreateGameSession](#), [DescribeGameSessionPlacement](#), atau [DescribePlayerSessions](#).

Tipe: String

Diperlukan: Ya

payload

Informasi yang ditentukan pengembang untuk dikomunikasikan ke server pada Realtime koneksi. Ini mencakup data acak yang mungkin digunakan untuk mekanisme kustom untuk masuk. Misalnya, payload dapat memberikan informasi otentikasi untuk diproses oleh skrip Realtime server sebelum mengizinkan klien untuk terhubung.

Jenis: byte array

Wajib: Tidak

RTMessage

Konten dan informasi pengiriman untuk pesan. Pesan harus menentukan pemain target atau grup target.

Daftar Isi

opCode

Kode operasi yang didefinisikan developer yang mengidentifikasi kejadian atau tindakan game, seperti gerakan pemain atau notifikasi server. Kode Op pesan memberikan konteks untuk muatan

data yang sedang diberikan. Pesan yang dibuat menggunakan `NewMessage()` sudah memiliki kode operasi yang ditetapkan, tetapi dapat diubah kapan saja.

Jenis: Integer

Wajib: Ya

`targetPlayer`

ID unik yang mengidentifikasi pemain yang merupakan penerima pesan yang dikirim. Target mungkin server itu sendiri (menggunakan ID server) atau pemain lain (menggunakan ID pemain).

Tipe: Integer

Wajib: Tidak

`targetGroup`

ID unik yang mengidentifikasi grup penerima pesan yang dikirim. Grup IDs didefinisikan oleh pengembang.

Tipe: Integer

Wajib: Tidak

`deliveryIntent`

Menunjukkan apakah akan mengirim pesan menggunakan koneksi TCP yang andal atau menggunakan saluran UDP cepat. Pesan yang dibuat menggunakan [NewMessage\(\)](#).

Jenis: `DeliveryIntent` enum

Nilai valid: FAST | RELIABLE

Wajib: Ya

`payload`

Isi pesan. Informasi ini terstruktur sesuai kebutuhan untuk diproses oleh client game berdasarkan kode operasi yang menyertainya. Ini mungkin berisi data status game atau informasi lain yang perlu dikomunikasikan antar client game atau antara client game dan server Realtime.

Jenis: Byte array

Wajib: Tidak

DataReceivedEventArgs

Data yang diberikan dengan callback [OnDataReceived\(\)](#).

Daftar Isi

pengirim

ID unik yang mengidentifikasi entitas (ID pemain atau ID server) yang mengirim pesan tersebut.

Jenis: Integer

Wajib: Ya

opCode

Kode operasi yang didefinisikan developer yang mengidentifikasi kejadian atau tindakan game, seperti gerakan pemain atau notifikasi server. Kode Op pesan memberikan konteks untuk muatan data yang sedang diberikan.

Jenis: Integer

Wajib: Ya

data

Isi pesan. Informasi ini terstruktur sesuai kebutuhan untuk diproses oleh client game berdasarkan kode operasi yang menyertainya. Ini mungkin berisi data status game atau informasi lain yang perlu dikomunikasikan antar client game atau antara client game dan server Realtime.

Jenis: Byte array

Wajib: Tidak

GroupMembershipEventArgs

Data yang diberikan dengan callback [OnGroupMembershipUpdated\(\)](#).

Daftar Isi

pengirim

ID unik mengidentifikasi pemain yang meminta pembaruan keanggotaan grup.

Jenis: Integer

Wajib: Ya

opCode

kode operasi yang didefinisikan developer yang mengidentifikasi kejadian atau tindakan game.

Jenis: Integer

Wajib: Ya

groupId

ID unik yang mengidentifikasi grup penerima pesan yang dikirim. Grup IDs didefinisikan oleh pengembang.

Jenis: Integer

Wajib: Ya

playerId

Daftar pemain IDs yang merupakan anggota saat ini dari grup yang ditentukan.

Jenis: himpunan integer

Wajib: Ya

Enum

Enum yang didefinisikan untuk SDK klien untuk Amazon GameLift Servers Realtime didefinisikan sebagai berikut:

ConnectionStatus

- **CONNECTED** — Klien game terhubung ke Realtime server hanya dengan koneksi TCP. Semua pesan, terlepas dari maksud pengirimannya, dikirim melalui TCP.
- **CONNECTED_SEND_FAST** — Klien game terhubung ke Realtime server dengan koneksi TCP dan UDP. Namun, kemampuan untuk menerima pesan melalui UDP belum diverifikasi; akibatnya, semua pesan yang dikirim ke client game menggunakan TCP.
- **CONNECTED_SEND_AND_RECEIVE_FAST** — Klien game terhubung ke server dengan koneksi TCP dan UDP. Realtime Client game dapat mengirim dan menerima pesan menggunakan TCP atau UDP.

- **CONNECTING** Game client telah mengirim permintaan koneksi dan Realtime server sedang memprosesnya.
- **DISCONNECTED_CLIENT_CALL** — Klien game terputus dari Realtime server sebagai tanggapan atas permintaan dari klien game. [Disconnect\(\)](#)
- **TERPUTUS** - Klien game terputus dari Realtime server karena alasan selain panggilan pemutusan klien.

ConnectionType

- **TLS12_RT_OVER_WSS_DTLS_** - Jenis koneksi aman.

Untuk digunakan dengan server Realtime yang berjalan pada Amazon GameLift Servers armada dengan sertifikat TLS yang dihasilkan. Apabila menggunakan sambungan aman, lalu lintas TCP dienkripsi menggunakan TLS 1.2, dan lalu lintas UDP dienkripsi menggunakan DTLS 1.2.

- **RT_OVER_WS_UDP_UNSECURED** — Jenis koneksi non-aman.
- **RT_OVER_WEBSOCKET** – Jenis koneksi non-aman. Nilai ini tidak lagi disukai.

DeliveryIntent

- **FAST** - Dikirimkan menggunakan saluran UDP.
- **ANDAL** - Dikirimkan menggunakan koneksi TCP.

Referensi skrip untuk Amazon GameLift ServersRealtime

Gunakan sumber daya ini untuk membangun logika kustom dalam Realtime skrip Anda.

Topik

- [Callback skrip untuk Amazon GameLift ServersRealtime](#)
- [Amazon GameLift ServersRealtimeantarmuka](#)

Callback skrip untuk Amazon GameLift ServersRealtime

Anda dapat memberikan logika khusus untuk menanggapi peristiwa dengan menerapkan callback ini dalam Realtime skrip Anda.

Init

Menginisialisasi Realtime server dan menerima antarmuka server Realtime.

Sintaksis

```
init(rtsession)
```

OnMessage

Dipanggil ketika pesan yang diterima dikirim ke server.

Sintaksis

```
onMessage(gameMessage)
```

onHealthCheck

Dipanggil untuk mengatur status kondisi sesi game. Secara default, status kondisi sehat (atau `true`). Callback ini dapat diimplementasikan untuk melakukan pemeriksaan kondisi kustom dan mengembalikan status.

Sintaksis

```
onHealthCheck()
```

onStartGameSesi

Dipanggil ketika sesi game baru dimulai, dengan objek sesi game yang dimasukkan.

Sintaksis

```
onStartGameSession(session)
```

onProcessTerminate

Dipanggil saat proses server dihentikan oleh layanan. Amazon GameLift Servers Hal ini dapat bertindak sebagai pemicu untuk keluar dengan bersih dari sesi game. Tidak perlu memanggil `processEnding()`.

Sintaksis

```
onProcessTerminate()
```

onPlayerConnect

Dipanggil ketika pemain meminta koneksi dan telah melewati validasi awal.

Sintaksis

```
onPlayerConnect(connectMessage)
```

onPlayerAccepted

Dipanggil ketika koneksi pemain diterima.

Sintaksis

```
onPlayerAccepted(player)
```

onPlayerDisconnect

Dipanggil ketika pemain terputus dari sesi game, baik dengan mengirimkan permintaan putuskan atau dengan cara lain.

Sintaksis

```
onPlayerDisconnect(peerId)
```

onProcessStarted

Dipanggil ketika proses server dimulai. Callback ini memungkinkan skrip untuk melakukan tugas-tugas kustom yang diperlukan untuk persiapan menjadi host sesi game.

Sintaksis

```
onProcessStarted(args)
```

onSendToPemain

Dipanggil ketika pesan diterima pada server dari satu pemain yang akan dikirim ke pemain lain. Proses ini berjalan sebelum pesan dikirim.

Sintaksis

```
onSendToPlayer(gameMessage)
```

onSendToKelompok

Dipanggil ketika pesan diterima pada server dari satu pemain yang akan dikirim ke grup. Proses ini berjalan sebelum pesan dikirim.

Sintaksis

```
onSendToGroup(gameMessage)
```

onPlayerJoinKelompok

Dipanggil ketika seorang pemain mengirimkan permintaan untuk bergabung dengan grup.

Sintaksis

```
onPlayerJoinGroup(groupId, peerId)
```

onPlayerLeaveKelompok

Dipanggil ketika seorang pemain mengirimkan permintaan untuk meninggalkan grup.

Sintaksis

```
onPlayerLeaveGroup(groupId, peerId)
```

Amazon GameLift Servers Realtime antarmuka

Ketika Amazon GameLift Servers Realtime skrip diinisialisasi, antarmuka ke Realtime server dikembalikan. Topik ini menjelaskan properti dan metode yang tersedia melalui antarmuka. Pelajari lebih lanjut tentang menulis Realtime skrip dan melihat contoh skrip terperinci di [Membuat Realtime skrip](#).

RealtimeAntarmuka menyediakan akses ke objek berikut:

- sesi
- pemain

- gameMessage
- konfigurasi

RealtimeObjek sesi

Gunakan metode ini untuk mengakses informasi terkait server dan melakukan tindakan terkait server.

getPlayers()

Mengambil daftar rekan IDs untuk pemain yang saat ini terhubung ke sesi permainan.
Mengembalikan himpunan objek pemain.

Sintaksis

```
rtSession.getPlayers()
```

broadcastGroupMembershipPerbarui ()

Memicu pengiriman daftar keanggotaan grup yang telah diperbarui kepada grup pemain.
Tentukan keanggotaan mana yang akan groupIdTo disiarkan (Broadcast) dan grup yang akan menerima pembaruan (targetGroupId). Grup IDs harus berupa bilangan bulat positif atau "-1" untuk menunjukkan semua kelompok. Lihat [Amazon GameLift ServersRealtimecontoh skrip](#) contoh grup yang ditentukan pengguna. IDs

Sintaksis

```
rtSession.broadcastGroupMembershipUpdate(groupIdToBroadcast, targetGroupId)
```

getServerId()

Mengambil pengidentifikasi ID unik rekan milik server, yang digunakan untuk merute pesan ke server.

Sintaksis

```
rtSession.getServerId()
```

getAllPlayersGroupId()

Mengambil ID grup untuk grup default yang berisi semua pemain yang saat ini terhubung ke sesi game.

Sintaksis

```
rtSession.getAllPlayersGroupId()
```

`processEnding()`

Memicu Realtime server untuk menghentikan server game. Fungsi ini harus dipanggil dari Realtime skrip untuk keluar dengan bersih dari sesi permainan.

Sintaksis

```
rtSession.processEnding()
```

`getGameSessionId ()`

Mengambil ID unik dari sesi game yang sedang berjalan.

Sintaksis

```
rtSession.getGameSessionId()
```

`getLogger()`

Mengambil antarmuka untuk pencatatan. Gunakan ini untuk mencatat pernyataan yang akan ditangkap dalam log sesi game Anda. Logger mendukung penggunaan pernyataan "info", "memperingatkan", dan "kesalahan". Sebagai contoh: `logger.info("<string>")`.

Sintaksis

```
rtSession.getLogger()
```

`sendMessage()`

Mengirim pesan, dibuat menggunakan `newTextGameMessage` atau `newBinaryGameMessage`, dari Realtime server ke penerima pemain menggunakan saluran UDP. Mengidentifikasi penerima menggunakan ID rekan pemain.

Sintaksis

```
rtSession.sendMessage(gameMessage, targetPlayer)
```

sendGroupMessage()

Mengirim pesan, dibuat menggunakan `newTextGameMessage` atau `newBinaryGameMessage`, dari Realtime server ke semua pemain dalam grup pemain menggunakan saluran UDP. Grup IDs harus berupa bilangan bulat positif atau "-1" untuk menunjukkan semua kelompok. Lihat [Amazon GameLift ServersRealtimecontoh skrip](#) contoh grup yang ditentukan pengguna. IDs

Sintaksis

```
rtSession.sendGroupMessage(gameMessage, targetGroup)
```

sendReliableMessage()

Mengirim pesan, dibuat menggunakan `newTextGameMessage` atau `newBinaryGameMessage`, dari Realtime server ke penerima pemain menggunakan saluran TCP. Mengidentifikasi penerima menggunakan ID rekan pemain.

Sintaksis

```
rtSession.sendReliableMessage(gameMessage, targetPlayer)
```

sendReliableGroupPesan ()

Mengirim pesan, dibuat menggunakan `newTextGameMessage` atau `newBinaryGameMessage`, dari Realtime server ke semua pemain dalam grup pemain menggunakan saluran TCP. Kelompok IDs yang harus berupa bilangan bulat positif atau "-1" untuk menunjukkan semua kelompok. Lihat [Amazon GameLift ServersRealtimecontoh skrip](#) contoh grup yang ditentukan pengguna. IDs

Sintaksis

```
rtSession.sendReliableGroupMessage(gameMessage, targetGroup)
```

newTextGamePesan ()

Membuat pesan baru yang berisi teks, yang akan dikirim dari server ke penerima pemain menggunakan `SendMessage` fungsi. Format pesan mirip dengan format yang digunakan dalam SDK klien untuk Realtime (lihat [RTMessage](#)). Mengembalikan objek `gameMessage`.

Sintaksis

```
rtSession.newTextGameMessage(opcode, sender, payload)
```

newBinaryGamePesan ()

Membuat pesan baru yang berisi data biner, yang akan dikirim dari server ke penerima pemain menggunakan SendMessage fungsi. Format pesan mirip dengan format yang digunakan dalam SDK klien untuk Realtime (lihat [RTMessage](#)). Mengembalikan objek gameMessage.

Sintaksis

```
rtSession.newBinaryGameMessage(opcode, sender, binaryPayload)
```

Objek pemain

Mengakses informasi yang berhubungan dengan pemain.

player.peerId

ID unik yang ditetapkan ke klien game saat terhubung ke Realtime server dan bergabung dengan sesi permainan.

pemain. playerSessionId

ID sesi pemain yang direferensikan oleh klien game saat terhubung ke Realtime server dan bergabung dengan sesi permainan.

Objek pesan game

Gunakan metode ini untuk mengakses pesan yang diterima oleh Realtime server. Pesan yang diterima dari client game memiliki struktur [RTMessage](#).

getPayloadAsTeks ()

Mendapatkan muatan pesan game sebagai teks.

Sintaksis

```
gameMessage.getPayloadAsText()
```

gameMessage.opcode

Kode operasi yang terkandung dalam pesan.

gameMessage.payload

Muatan yang terkandung dalam pesan. Mungkin teks atau biner.

`gameMessage.sender`

ID rekan client game yang mengirim pesan.

`gameMessage.reliable`

Boolean yang menunjukkan apakah pesan dikirim melalui TCP (betul) atau UDP (salah).

Objek konfigurasi

Objek konfigurasi dapat digunakan untuk menimpa konfigurasi default.

`configuration.maxPlayers`

Jumlah maksimum koneksi klien/server yang dapat diterima oleh RealTimeServers.

Default-nya adalah 32.

`konfigurasi.pingIntervalTime`

Interval waktu dalam milidetik yang akan dicoba server untuk mengirim ping ke semua client yang terhubung untuk memverifikasi koneksi yang sehat.

Defaultnya adalah 3000ms.

Acara penempatan sesi permainan

Amazon GameLift Servers memancarkan acara untuk setiap permintaan penempatan sesi game saat diproses. Anda dapat memublikasikan peristiwa ini ke topik Amazon SNS, seperti yang dideskripsikan dalam [Atur notifikasi kejadian untuk penempatan sesi game](#). Peristiwa ini juga dipancarkan ke Amazon CloudWatch Events dalam waktu dekat dan dengan upaya terbaik.

Topik ini menjelaskan struktur peristiwa penempatan sesi game dan memberikan contoh untuk setiap jenis peristiwa. Untuk informasi selengkapnya tentang status permintaan penempatan sesi game, lihat [GameSessionPlacement](#) di Amazon GameLift Servers Referensi API.

Sintaksis peristiwa penempatan

Peristiwa direpresentasikan sebagai objek JSON. Struktur acara sesuai dengan pola CloudWatch Peristiwa, dengan bidang tingkat atas yang serupa dan detail khusus layanan.

Bidang tingkat atas mencakup hal-hal berikut (lihat [Pola peristiwa](#) untuk lebih detailnya):

versi

Bidang ini selalu diatur ke 0 (nol).

id

Pengenal pelacakan unik untuk peristiwa tersebut.

jenis-detail

Nilainya selalu `GameLift Queue Placement Event`.

sumber

Nilainya selalu `aws.gamelift`.

akun

AWS Akun yang digunakan untuk mengelola Amazon GameLift Servers.

Waktu

Stempel waktu peristiwa.

region

AWS Wilayah tempat permintaan penempatan sedang diproses. Ini adalah Wilayah di mana antrean sesi game digunakan berada.

sumber daya

Nilai ARN dari antrean sesi game yang memproses permintaan penempatan.

PlacementFulfilled

Permintaan penempatan telah berhasil dipenuhi. Sesi game baru telah dimulai dan sesi pemain baru telah dibuat untuk setiap pemain yang tercantum dalam permintaan penempatan sesi game. Informasi koneksi pemain tersedia.

Detail sintaks:

placementId

Pengidentifikasi unik yang ditetapkan untuk permintaan penempatan sesi game.

Port

Nomor port untuk sesi game baru.

gameSessionArn

Pengenal ARN untuk sesi game baru.

ipAddress

Alamat IP dari sesi game.

dnsName

Pengenal DNS yang ditetapkan untuk instans yang menjalankan sesi game baru. Format nilai berbeda tergantung pada apakah instans yang menjalankan sesi game diaktifkan TLS. Saat menghubungkan ke sesi game di armada yang mendukung TLS, pemain harus menggunakan nama DNS, bukan alamat IP.

Armada yang mendukung TLS:<unique identifier>.<region identifier>.amazongamelift.com.

Non-TLS-enabled armada:ec2-<unique identifier>.compute.amazonaws.com.

startTime

Stempel waktu menunjukkan kapan permintaan ini ditempatkan dalam antrian.

endTime

Stempel waktu menunjukkan kapan permintaan ini terpenuhi.

gameSessionRegion

AWS Wilayah armada yang menjadi tuan rumah sesi permainan. Ini sesuai dengan token wilayah diGameSessionArn.

placedPlayerSessions

Kumpulan sesi pemain yang telah dibuat untuk setiap pemain dalam permintaan penempatan sesi game.

Contoh

```
{
```

```

"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

Permintaan penempatan dibatalkan dengan panggilan ke GameLift layanan [StopGameSessionPlacement](#).

Detail:

placementId

Pengenal unik yang ditetapkan untuk permintaan penempatan sesi game.

startTime

Stempel waktu menunjukkan kapan permintaan ini ditempatkan dalam antrian.

endTime

Stempel waktu menunjukkan kapan permintaan ini dibatalkan.

Contoh

```
{
  "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

Penempatan sesi game tidak berhasil diselesaikan sebelum batas waktu antrean berakhir. Permintaan penempatan dapat dikirim ulang sesuai kebutuhan.

Detail:

placementId

Pengenal unik yang ditetapkan untuk permintaan penempatan sesi game.

startTime

Stempel waktu menunjukkan kapan permintaan ini ditempatkan dalam antrean.

endTime

Stempel waktu menunjukkan kapan permintaan ini dibatalkan.

Contoh

```
{
  "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 tidak dapat memenuhi permintaan sesi permainan. Hal ini umumnya disebabkan oleh kesalahan internal yang tidak terduga. Permintaan penempatan dapat dikirim ulang sesuai kebutuhan.

Detail:

`placementId`

Pengenal unik yang ditetapkan untuk permintaan penempatan sesi game.

`startTime`

Stempel waktu menunjukkan kapan permintaan ini ditempatkan dalam antrian.

`endTime`

Stempel waktu menunjukkan ketika permintaan ini gagal.

Contoh

```
{
  "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 ServersVersi AMI

Tabel berikut mengidentifikasi gambar mesin Amazon terbaru (AMIs) yang Amazon GameLift Servers digunakan untuk EC2 hosting terkelola. Seperti yang dijelaskan dalam [Analisis konfigurasi dan kerentanan di Amazon GameLift Servers](#), Anda harus secara teratur membuat EC2 armada Amazon GameLift Servers terkelola baru untuk menerapkan pembaruan versi AMI terbaru.

AMIs untuk digunakan dengan server SDK untuk 5+ Amazon GameLift Servers

Amazon GameLift Servers menggunakan yang berikut ini AMIs untuk meng-host server game yang terintegrasi dengan server SDK untuk Amazon GameLift Servers versi 5.

Amazon GameLift Serversgambar	Arsitektur	Patch terbaru	AWS Gambar dasar
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 (catatan rilis)

Amazon GameLift Serversgambar	Arsitektur	Patch terbaru	AWS Gambar dasar
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 (catatan rilis)
Windows 2016 BASE_AMI_WINDOWS_2016_SDK_5	x86	2025-04-22	Windows_Server-2016-English-Full-Base-2025.03.12 (catatan rilis)
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 (catatan rilis)
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 (catatan rilis)

AMIs untuk digunakan dengan server SDK untuk 4 Amazon GameLift Servers

Amazon GameLift Serversmenggunakan yang berikut ini AMIs untuk meng-host server game yang terintegrasi dengan server SDK untuk Amazon GameLift Servers versi 4 atau lebih lama.

Amazon GameLift Serversgambar	Arsitektur	Patch terbaru	AWS Gambar dasar
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 (catatan rilis)
Windows 2016	x86	2025-04-22	Windows_Server-2016-English-Full-Base-2025.03.12 (catatan rilis)

Amazon GameLift Serversgambar	Arsitektur	Patch terbaru	AWS Gambar dasar
BASE_AMI_WINDOWS_2016_SDK_4			

Untuk informasi selengkapnya, lihat sumber daya berikut:

- [Catatan rilis Amazon Linux 2023](#)
- [Catatan rilis Amazon Linux 2](#)
- [AWS Riwayat versi Windows AMI](#)
- [Deskripsi Layanan Pembaruan Perangkat Lunak dan Layanan Pembaruan Windows Server perubahan konten untuk tahun 2024](#)

Titik akhir dan kuota Amazon GameLift Servers

- [Untuk titik akhir layanan dan titik akhir suar yang dapat Anda gunakan untuk terhubung secara terprogram dengan layanan, lihat titik akhir layanan. Amazon GameLift Servers](#)
- Untuk Amazon GameLift Servers kuota tentang penggunaan sumber daya layanan atau operasi per AWS akun, lihat [kuota Amazon GameLift Servers layanan](#). Untuk informasi lebih lanjut tentang kuota dan cara meminta kenaikan, lihat [kuota AWS layanan](#). Anda dapat meminta peningkatan kuota dengan menggunakan Amazon GameLift Servers konsol.

Amazon GameLift ServersSuar ping UDP

UDP ping beacon menyediakan cara untuk mengukur latensi jaringan antara perangkat pemain dan lokasi hosting. Amazon GameLift Servers Dengan ping beacon, Anda dapat mengumpulkan data latensi yang akurat untuk membuat keputusan berdasarkan informasi tentang penempatan server game dan meningkatkan perjodohan pemain berdasarkan persyaratan latensi.

Cara kerja ping beacon UDP

Amazon GameLift Servers menyediakan endpoint UDP tetap (ping beacon) di setiap lokasi hosting tempat Anda dapat menyebarkan server game. Karena sebagian besar server game berkomunikasi menggunakan UDP, mengukur latensi dengan ping beacon UDP memberikan hasil yang lebih

akurat daripada menggunakan ping ICMP. Perangkat jaringan sering menangani paket ICMP secara berbeda dari paket UDP, yang dapat menyebabkan pengukuran latensi yang tidak mencerminkan kinerja sebenarnya yang akan dialami pemain Anda.

Dengan suar ping UDP, klien game Anda dapat mengirim pesan UDP ke titik akhir ini dan menerima respons asinkron, memberi Anda pengukuran latensi yang lebih mewakili kondisi lalu lintas game aktual antara perangkat pemain dan lokasi hosting potensial. Titik akhir bersifat permanen dan tetap tersedia selama Amazon GameLift Servers mendukung hosting game di lokasi tersebut.

Kasus penggunaan umum untuk suar ping UDP

Anda dapat menggunakan ping beacon UDP dalam beberapa cara untuk mengoptimalkan pengalaman jaringan game Anda.

Memilih lokasi hosting yang optimal

Kumpulkan data latensi di berbagai wilayah geografis untuk mengidentifikasi lokasi utama dan cadangan terbaik untuk hosting server game untuk basis pemain Anda.

Menempatkan sesi permainan berdasarkan latensi pemain

Sertakan data latensi pemain saat meminta sesi permainan baru untuk membantu memilih lokasi yang memberikan pengalaman latensi terendah.

Mengoptimalkan perjodohan berdasarkan latensi

Berikan data latensi pemain saat meminta perjodohan untuk membantu mencocokkan pemain dengan profil latensi serupa dan menempatkan sesi permainan di lokasi optimal untuk pemain yang cocok.

Note

Saat membuat permintaan perjodohan, Anda tidak boleh memberikan informasi latensi ke lokasi di mana Anda tidak memiliki armada. Jika Anda melakukannya, Amazon GameLift Servers mungkin mencoba menempatkan sesi permainan di lokasi di mana tidak ada kapasitas armada, yang mengakibatkan kegagalan permintaan perjodohan.

Mendapatkan titik akhir suar

Untuk mengambil domain ping beacon dan informasi port untuk Amazon GameLift Servers lokasi, gunakan operasi API. [ListLocations](#) Kumpulan lokasi yang dikembalikan oleh API ini bergantung pada yang Wilayah AWS Anda tentukan saat memanggilnya (atau Wilayah default jika Anda tidak menentukannya). Saat Anda menelepon dari:

- Wilayah asal armada yang mendukung multi-lokasi: API mengembalikan informasi untuk semua lokasi hosting
- Wilayah asal armada yang mendukung satu lokasi: API mengembalikan informasi untuk lokasi tersebut

Perhatikan bahwa jika Anda memanggil API ini menggunakan lokasi yang hanya dapat berupa lokasi jarak jauh dalam armada multi-lokasi, API akan menampilkan kesalahan karena jenis lokasi tersebut tidak memiliki titik akhir layanan.

Lihat tabel lokasi yang didukung [AWS Lokasi yang didukung](#) untuk mengidentifikasi Wilayah asal yang mendukung armada tunggal dan multi-lokasi.

Contoh

```
aws gamelift list-locations --region ap-northeast-2
```

Ini Wilayah AWS mendukung armada multi-lokasi, oleh karena itu beberapa lokasi akan dikembalikan. Berikut adalah contoh dari salah satu nilai yang dikembalikan:

```
[...]  
{  
  "LocationName": "ap-northeast-1",  
  "PingBeacon": {  
    "UDPEndpoint": {  
      "Domain": "gamelift-ping.ap-northeast-1.api.aws",  
      "Port": 7770  
    }  
  }  
}
```

 Important

Cache informasi ping beacon daripada menelepon `ListLocations` sebelum setiap pengukuran latensi. Informasi domain dan port bersifat statis dan API tidak dirancang untuk permintaan volume tinggi.

Menerapkan pengukuran latensi

Ikuti praktik terbaik ini saat menerapkan pengukuran latensi menggunakan suar ping UDP:

1. Simpan informasi ping beacon menggunakan salah satu pendekatan ini:

- Hardcode titik akhir di klien game Anda.
- Cache informasi di backend game Anda.
- Menerapkan mekanisme pembaruan berkala (harian/mingguan) untuk menyegarkan informasi.

2. Kirim pesan ping UDP:

- Masukkan apa pun yang Anda inginkan di badan pesan, selama tidak kosong, dan Anda menyimpan pesan di bawah ukuran maksimum 300 byte.
- Perhatikan batas tarif berikut untuk setiap lokasi:
 - 3 transaksi per detik (TPS) per alamat IP pengirim unik dan kombinasi port
 - 1000 TPS per alamat IP pengirim unik

3. Hitung latensi:

- Kirim beberapa ping ke setiap lokasi untuk menghitung latensi rata-rata.
- Pertimbangkan untuk mengirim ping bersamaan ke beberapa lokasi untuk hasil yang lebih cepat.
- Gunakan logika coba lagi sesuai kebutuhan untuk mengirim paket baru untuk paket apa pun yang tidak dikembalikan dalam waktu singkat (biasanya 1 - 3 detik), karena UDP tidak memiliki jaminan pengiriman 100%.
- Hitung perbedaan waktu antara mengirim pesan dan menerima respons.
- Jika sebagian besar ping UDP ke lokasi secara konsisten tidak mendapatkan respons kembali, hitung latensi menggunakan ping ICMP ke titik akhir [Amazon GameLift Serverslayanan](#) standar kami sebagai fallback.

 Tip

Kami menyarankan Anda menyertakan port 7770 di mana pun Anda mendokumentasikan daftar port yang harus dibuka pemain Anda di jaringan lokal mereka. Ini adalah alasan lain mengapa Anda harus memiliki fallback untuk mengukur latensi (menggunakan ICMP, misalnya) jika port ini diblokir.

Contoh kode

Berikut adalah beberapa contoh sederhana yang menunjukkan cara mengirim ping UDP dan menghitung latensi.

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()
```


Versi sebelumnya

Amazon S3
Klik
Uraian

10
Kesatuan
+
Tidak
Nyata

052
atau
yang
lebih
baru

052
NET
yang
lebih
baru

052
atau
yang
lebih
baru

051
atau
yang
lebih
baru

051
atau

Amazon S3

Amazon

Amazon

Amazon

Amazon

Amazon

+

Tidak

Nyata

Yang

Lebih

Baru

Baru

170

atau

Yang

Lebih

Baru

Baru

151

atau

Yang

Lebih

Baru

Baru

151

atau

Yang

Lebih

Baru

Baru

151

atau

Yang

Lebih

Baru

Baru

Amazon S3
Amazon
Unitas

20
kesatuan
+
Tidak
Nyata

20251
atau
yang
lebih
baru

20251
atau
yang
lebih
baru

20231
atau
yang
lebih
baru

20230
atau
yang
lebih
baru

baru

Amazon S3
Amazon
Unitas

2%
kesatuan
+
Tidak
Nyata

[502406-1](#)
atau
yang
lebih
baru

[502405-2](#)
atau
yang
lebih
baru

[502404-2](#)
atau
yang
lebih
baru

[502404-1](#)
atau
yang
lebih
baru

Amazon GameLift
Amazon
Unitas

20
kesatuan
+
Tidak
Nyata

[5142002-0](#)
atau
yang
lebih
baru

[5142001-3](#)
atau
yang
lebih
baru

[5142002-0](#)
NET
yang
lebih
baru

[5142003-2](#)
atau
yang
lebih
baru

Amazon GameLift
Amazon
Unitas

20
kesatuan
+
Tidak
Nyata

[801220-2](#)
atau
yang
lebih
baru

[801205-0](#)
atau
yang
lebih
baru

[801303-2](#)
atau
yang
lebih
baru

[801303-1](#)
atau
yang
lebih
baru

Amazon S3
Amazon
Unitas

20
kesatuan
+
Tidak
Nyata

[80120-0](#)
atau
yang
lebih
baru

[80120-2](#)
atau
yang
lebih
baru

[80120-1-2](#)
atau
yang
lebih
baru

[80120-1-1](#)
atau
yang
lebih
baru

Amazon S3
Amazon
Unitas

20
kesatuan
+
Tidak
Nyata

[82120-1](#)
atau
yang
lebih
baru

[82120-2](#)
atau
yang
lebih
baru

[82120-1](#)
atau
yang
lebih
baru

[82120-0](#)
atau
yang
lebih
baru

Amazon S3
Amazon
Unitas

2
kesatuan
+
Tidak
Nyata

[B74900-1](#)
atau
yang
lebih
baru

[B74900-1](#)
atau
yang
lebih
baru

[B74900-2](#)
atau
yang
lebih
baru

[B74900-0](#)
atau
yang
lebih
baru

Windows
Linux
Ubuntu

1/2
Kesatuan
+Tidak
Nyata

[B3007-0](#)
atau
yang
lebih
baru

[B3004-2](#)
atau
yang
lebih
baru

[B3003-0](#)
atau
yang
lebih
baru

[B3002-0](#)
atau
yang
lebih
baru

Amazon S3
Amazon
Unitas

2/6
kesatuan
+
Tidak
Nyata

[B33012-1](#)
atau
yang
lebih
baru

[B22109-2](#)
atau
yang
lebih
baru

[B221106-1](#)
atau
yang
lebih
baru

[B221105-1](#)
atau
yang
lebih
baru

Amazon S3
Amazon
Unitas

2%
kesatuan
+
Tidak
Nyata

[B.2.3.102-1](#)
atau
yang
lebih
baru

[B.2.3.102-0](#)
atau
yang
lebih
baru

[B.1.7.309-0](#)
atau
yang
lebih
baru

[B.1.7.308-1](#)
atau
yang
lebih
baru

Amazon S3
Amazon
Unitas

2
kesatuan
+
Tidak
Nyata

[BC17505-1](#)
atau
yang
lebih
baru

[BC17504-1](#)
atau
yang
lebih
baru

[BC17502-2](#)
atau
yang
lebih
baru

[BC16011-1](#)
atau
yang
lebih
baru

3.1.0-1

3.1.0-1

3.1.0-1

3.1.0-1

3.1.0-1

3.1.0-1

3.1.0-1

3.1.0-1

3.1.0-1

3.1.0-1

3.1.0-1

3.1.0-1

3.1.0-1

3.1.0-1

3.1.0-1

3.1.0-1

3.1.0-1

3.1.0-1

3.1.0-1

3.1.0-1

3.1.0-1

3.1.0-1

3.1.0-1

Catatan rilis

Catatan rilis berikut dalam urutan kronologis, dengan pembaruan terbaru terdaftar terlebih dahulu. Amazon GameLift Servers pertama kali dirilis pada tahun 2016. Untuk catatan rilis pada tanggal lebih awal dari yang tercantum di sini, lihat link tanggal rilis di [Versi SDK](#).

24 Juni 2025: Amazon GameLift Servers meluncurkan suar ping UDP untuk membantu mengukur latensi jaringan waktu nyata untuk game multipemain

Versi SDK yang diperbarui:

- AWS SDK 1.11.595

Amazon GameLift Servers rilis untuk ketersediaan umum satu set titik akhir tetap yang disebut UDP ping beacon untuk membantu Anda mengukur latensi secara akurat antara perangkat pemain dan lokasi server game. Titik akhir ping beacon UDP tersedia di semua Wilayah AWS Global dan Zona Lokal yang didukung oleh Amazon GameLift Servers, kecuali Wilayah Tiongkok. AWS

Sebagian besar game multipemain menggunakan UDP (User Datagram Protocol) sebagai protokol transmisi paket utama mereka karena manfaat kinerjanya untuk game real-time. Memahami dan mengoptimalkan latensi jaringan sangat penting untuk memberikan pengalaman pemain terbaik. UDP ping beacon menyediakan cara yang konsisten dan andal untuk mengukur latensi paket UDP aktual antara pemain dan server game, membantu Anda membuat keputusan yang lebih baik tentang player-to-server pencocokan dan penempatan sesi permainan.

[ListLocations](#) API telah diperluas untuk menyertakan domain endpoint dan informasi port sebagai bagian dari daftar lokasi yang dikembalikan, sehingga mudah untuk mengakses endpoint secara terprogram.

Klien game Anda dapat mengirim pesan UDP ke titik akhir ini dan menerima respons asinkron dengan data yang sama, memberi Anda pengukuran latensi yang lebih mewakili kondisi lalu lintas game aktual antara perangkat pemain dan lokasi hosting potensial. Endpoint ini bersifat permanen dan tetap tersedia selama Amazon GameLift Servers mendukung hosting game di lokasi tersebut.

Pelajari lebih lanjut:

- [Suar ping UDP](#), Panduan Amazon GameLift Servers Pengembang
- [ListLocations](#), Referensi Amazon GameLift Servers API

24 Juni 2025: Amazon GameLift Servers menambahkan dukungan lokasi untuk Thailand dan Malaysia

Dengan hosting Amazon GameLift Servers terkelola, Anda sekarang dapat menyebarkan sumber daya server game di Bangkok, Thailand, dan Kuala Lumpur, Malaysia, memperluas jangkauan game

Anda ke pemain di seluruh Asia Tenggara. Wilayah baru ini membantu mengurangi latensi dan meningkatkan pengalaman bermain game bagi pemain di area tersebut.

Berikut ini Wilayah AWS tersedia sebagai lokasi terpencil untuk armada multi-lokasi. Untuk mulai menyelenggarakan sesi permainan di lokasi-lokasi ini, tambahkan mereka sebagai lokasi terpencil ke armada multi-lokasi baru atau yang sudah ada. Dengan armada multi-lokasi, Anda dapat langsung mengelola kapasitas hosting di setiap lokasi.

- Asia Pasifik (Thailand) (ap-southeast-7)
- Asia Pasifik (Malaysia) (ap-southeast-5)

Ini Wilayah AWS tidak diaktifkan secara default untuk AWS akun. Anda dapat memilih setiap Wilayah sebelum Anda dapat menerapkan Amazon GameLift Servers sumber daya di sana.

Pelajari lebih lanjut:

- [Amazon GameLift Servers lokasi layanan](#), Panduan Pengembang Amazon GameLift Servers Hosting
- [Perbarui lokasi armada](#), Panduan Pengembang Amazon GameLift Servers Hosting
- [Mengaktifkan atau menonaktifkan AWS Wilayah di akun Anda](#), Panduan Referensi Manajemen AWS Akun.

29 Mei 2025: Amazon GameLift Servers merilis versi baru SDK server dan plugin untuk Unreal Engine dan Unity

Versi SDK yang diperbarui:

- [SDK server C ++, versi 5.3.0](#)
- [SDK server C #, versi 5.3.0](#)
- [Go server SDK](#), versi 5.3.0
- [Server SDK untuk Unreal, versi 5.3.0](#)
- [Server SDK for Unity](#), versi 5.3.0

Diperbarui versi plugin:

- [Plugin untuk Unreal](#), versi 3.0.0

- [Plugin untuk Unity](#), versi 3.1.0

Versi baru SDK server untuk C++, C #, Go, Unreal dan Unity, serta versi plugin baru untuk Unreal Engine dan Unity, sekarang open source. Semuanya tersedia di [Amazon GameLift Servers GitHub organisasi](#). Untuk informasi pembaruan terperinci, lihat catatan rilis dan readmes di setiap repositori.

Pembaruan SDK server kunci:

- Peningkatan validasi sisi klien dan respons kesalahan di semua server. SDKs
- Fungsi [OnProcessTerminate](#) callback sekarang memiliki logika default untuk mengakhiri proses server game.
- Fungsi ini [InitSDK\(\)](#) sekarang menggunakan token idempotensi untuk mendukung beberapa percobaan ulang.
- [OnUpdateGameSession](#) Callback sekarang dapat meneruskan data untuk autoBackfillMode.

Pembaruan plugin utama:

- Plugin untuk Unreal Engine sekarang memiliki proses instalasi dan penyiapan yang lebih efisien, dengan lebih banyak otomatisasi dan lebih sedikit prasyarat (CMake, OpenSSL, dan rantai alat kompilasi silang Unreal).
- Plugin untuk Unreal Engine telah meningkatkan pengalaman UI untuk EC2 alur kerja Terkelola, termasuk dukungan untuk spasi di klien game dan jalur pembuatan server. Selain itu, Anda sekarang dapat menambahkan argumen baris perintah saat meluncurkan klien game dari Editor.
- Plugin untuk Unreal Engine sekarang mendukung server ARM yang dibangun. UE5

Pelajari lebih lanjut:

- [Amazon GameLift Serversplugin untuk Unreal Engine](#), Panduan Amazon GameLift Servers Pengembang
- [Amazon GameLift Serversplugin untuk Unity \(server SDK 5.x\)](#), Panduan Amazon GameLift Servers Pengembang
- [Server SDK 5.x untuk Amazon GameLift Servers](#), Panduan Amazon GameLift Servers Pengembang
- [Integrasikan Amazon GameLift Servers ke dalam proyek Unreal Engine](#), Panduan Amazon GameLift Servers Pengembang

- [Integrasikan Amazon GameLift Servers ke dalam proyek Unity](#), Panduan Amazon GameLift Servers Pengembang

24 April 2025: Server C# SDK 5 untuk Amazon GameLift Servers saat ini mendukung .NET 8

Versi SDK yang diperbarui:

- C# Server SDK, versi 5.2.1

Untuk pengembang game yang menggunakan C #, Anda sekarang dapat menggunakan .NET 8 sebagai kerangka kerja target untuk Amazon GameLift Servers proyek Anda. Dengan .NET 8, Anda dapat memanfaatkan peningkatan kinerja, termasuk kompilasi yang ditingkatkan just-in-time (JIT), pengoptimalan penggunaan memori, dan waktu startup yang lebih cepat. Jika saat ini Anda menggunakan .NET 6, sebaiknya Anda merencanakan migrasi ke .NET 8, termasuk memperbarui SDK server C# Anda ke versi terbaru. Microsoft mengumumkan dukungan .NET 8, dengan patch keamanan dan pembaruan teknis, hingga November 2026.

[Unduh versi terbaru SDK server C # untuk Amazon GameLift Servers](#)

Pelajari lebih lanjut:

- [Dapatkan alat Amazon GameLift Servers pengembangan](#), Panduan Amazon GameLift Servers Pengembang

27 Maret 2025: Amazon GameLift Servers memperluas dukungan untuk keluarga instans generasi berikutnya EC2

Versi SDK yang diperbarui:

- AWS SDK 1.11.535

Anda sekarang dapat menyempurnakan hosting server game Anda Amazon GameLift Servers dengan memilih dari lebih banyak pilihan EC2 instans Amazon di seluruh keluarga instans generasi ke-5 hingga ke-8. Setiap EC2 generasi baru menawarkan kemajuan dalam EC2 komputasi, memori, dan jaringan, dengan instans generasi ke-8 menghadirkan instans berbasis Graviton4 dan Intel Xeon mutakhir AWS . Instance generasi berikutnya tersedia dalam keluarga instance berikut:

- [Tujuan umum \(seri-M\)](#)
- [Komputasi dioptimalkan](#) (C-series)
- [Memori dioptimalkan](#) (R-series)

Anda juga dapat memilih varian yang menawarkan penyimpanan lokal (d), jaringan yang disempurnakan (n), dan arsitektur prosesor tertentu (Intel/AMD/Graviton – i/a/g). Instans generasi berikutnya tersedia di semua Wilayah AWS yang didukung oleh Amazon GameLift Servers, kecuali Wilayah Tiongkok AWS. Untuk detail selengkapnya, lihat [Amazon GameLift Servers lokasi layanan](#).

Gunakan tipe instance baru ini dengan EC2 armada Amazon GameLift Servers terkelola dan armada kontainer terkelola. Saat mengalihkan hosting game Anda yang ada ke jenis instans baru (arsitektur yang sama), cukup gunakan armada baru dengan semua pengaturan konfigurasi tidak berubah kecuali jenis instance.

Pelajari lebih lanjut:

- [Amazon GameLift Servers Harga Instance](#)
- [CreateFleetAmazon GameLift Servers](#) Referensi API
- [CreateContainerFleet](#), Referensi Amazon GameLift Servers API

11 Maret 2025: Amazon GameLift Servers meluncurkan Game Server Wrapper baru untuk orientasi cepat

Pembungkus server game baru untuk Amazon GameLift Servers secara signifikan mengurangi waktu yang diperlukan untuk mendapatkan server game yang di-host. Amazon GameLift Servers Tanpa perlu perubahan kode, Anda dapat menggunakan pembungkus untuk menambahkan fungsionalitas manajemen sesi permainan dasar ke game Anda dan menyebarkannya ke armada Amazon GameLift Servers Anywhere, armada terkelola, atau EC2 armada kontainer terkelola. Alat ini sangat ideal untuk melakukan evaluasi langsung Amazon GameLift Servers fitur, menggunakan server game Anda sendiri atau server dari contoh game. Hal ini juga berguna untuk cepat menyebarkan iterasi server game, seperti untuk prototyping cepat atau pengujian.

Dengan fitur manajemen sesi permainan dasar, server game Anda dapat menginisialisasi koneksi dengan Amazon GameLift Servers layanan, menanggapi permintaan untuk memulai dan menghentikan sesi permainan, dan mematikan saat sesi permainan selesai.

[Unduh pembungkus server game dari GitHub](#). Lihat readme tentang GitHub cara menginstal dan menggunakan pembungkus dengan proyek game Anda.

6 Maret 2025: Amazon GameLift sekarang Amazon GameLift Servers dan Amazon Streams GameLift

Amazon GameLift adalah layanan yang dikelola sepenuhnya yang didedikasikan untuk membantu pengembang membangun, menskalakan, dan menghadirkan game paling menuntut di dunia. Dengan rilis ketersediaan umum Amazon GameLift Streams, Amazon GameLift sekarang menawarkan server game skala tinggi dan kemampuan streaming gameplay yang mulus.

[Amazon GameLift Servers](#) memberi pengembang game kemampuan untuk menyebarkan, mengoperasikan, dan menskalakan server game khusus. Anda dapat menyebarkan server game berkinerja tinggi di cloud dalam hitungan menit dan meningkatkan dan menurunkan skala untuk memenuhi permintaan pemain. Dibangun di atas lingkungan komputasi yang AWS terbukti, Amazon GameLift Servers mendukung 100 juta pemain bersamaan dalam satu game, 100 ribu pemain menambahkan per detik, dan 9 ribu instance komputasi baru per menit. Dan dengan keamanan tingkat perusahaan, perijinan untuk kerumunan terbesar, dan pay-as-you-go fleksibilitas, ini membantu Anda memulai apakah Anda sedang mengerjakan ide game baru atau menjalankan game dengan jutaan pemain.

[Amazon GameLift Streams](#) membantu pengembang game menghadirkan pengalaman streaming game hingga resolusi 1080p dan 60 frames-per-second (fps) tanpa jeda yang terlihat di seluruh perangkat termasuk iOS, Android, FireOS, dan PCs untuk gamer. Dengan menggunakan satu AWS penawaran, penayang dapat menyebarkan konten game mereka dalam hitungan menit, tanpa modifikasi, ke instance GPU berbasis cloud yang dikelola sepenuhnya dan mengirimkannya melalui AWS Network Backbone langsung ke perangkat end-player mana pun dengan browser web. Pemain dapat mulai bermain game hanya dalam beberapa detik, tanpa menunggu unduhan atau pemasangan dan memberikan pengalaman bermain game yang hampir tidak dapat dibedakan dari bermain game secara lokal di PC atau konsol game.

14 Januari 2025: Amazon GameLift Servers rilis memperluas dukungan untuk prioritas penempatan sesi game

Versi SDK yang diperbarui:

- AWS SDK 1.11.485

Menanggapi umpan balik pelanggan, kami merilis fungsionalitas baru yang memungkinkan Anda memprioritaskan lokasi untuk permintaan penempatan sesi game individual. Untuk antrian Anda yang dikonfigurasi untuk memprioritaskan penempatan berdasarkan lokasi, Anda sekarang dapat memberikan daftar lokasi prioritas yang disesuaikan dengan setiap permintaan penempatan.

Fitur baru ini memungkinkan pelanggan secara dinamis mengubah prioritas lokasi untuk setiap permintaan penempatan sesuai kebutuhan. Fleksibilitas tambahan berarti Anda dapat merespons kondisi yang berubah dengan lebih baik, seperti lokasi pemain, beban armada, atau kesehatan server. Ini juga dapat mendukung pelanggan yang ingin menyesuaikan lebih lanjut bagaimana lokasi penempatan dipilih.

Pelajari lebih lanjut:

- [Prioritaskan penempatan sesi game, Panduan Pengembang Amazon GameLift Servers](#)
- [StartGameSessionPlacement](#), Referensi Amazon GameLift Servers API

2 Januari 2025: Amazon GameLift Servers merilis dukungan baru untuk mengakhiri sesi game

Versi SDK yang diperbarui:

- AWS SDK 1.11.477

Menanggapi umpan balik pelanggan, kami merilis fungsionalitas baru yang memungkinkan Anda lebih mudah menghentikan sesi permainan individual. Dengan rilis ini, Anda sekarang dapat mengakhiri sesi game langsung di Amazon GameLift Servers konsol atau dengan menggunakan AWS CLI atau AWS SDK untuk. Amazon GameLift Servers

Fitur baru ini membahas kebutuhan untuk menyelesaikan sesi permainan yang tetap aktif tetapi dalam keadaan buruk, yang mencegah sumber daya komputasi menghosting sesi game baru. Sebelumnya, pelanggan diminta untuk mengakses komputasi dari jarak jauh untuk menghentikan sesi permainan secara manual.

Anda memiliki dua metode penghentian untuk dipilih. Metode pertama mencoba untuk mengakhiri sesi permainan dengan anggun menggunakan urutan shutdown kustomnya, yang mungkin mencakup tindakan untuk memberi tahu pemain dan menyelesaikan data game. Metode kedua memaksa proses server berhenti, yang segera mengakhiri sesi permainan. Metode kedua ini memastikan bahwa sesi permainan berakhir bahkan ketika proses server tidak merespons.

Pelajari lebih lanjut:

- [Matikan sesi permainan menggunakan Amazon GameLift Servers konsol](#), Panduan Amazon GameLift Servers Pengembang
- [TerminateGameSession](#), Referensi Amazon GameLift Servers API

19 Desember 2024: Amazon GameLift Servers merilis dukungan plugin mesin game untuk Kontainer Terkelola

Diperbarui versi plugin:

Amazon GameLift Serversplugin untuk Unreal Engine, versi 2.0.0

- Diupgrade untuk mendukung server C ++ SDK 5.2.0 dengan dukungan kontainer terkelola.
- Menambahkan dukungan untuk Unreal Engine 5.4 dan 5.5.

Amazon GameLift Serversplugin untuk Unity, versi 3.0.0

- Diupgrade untuk mendukung server C ++ SDK 5.2.0 dengan dukungan kontainer terkelola.
- Support untuk Unity 2021.3 LTS dan 2022.3 LTS untuk Windows dan Mac OS.

Amazon GameLift ServersPlugin untuk mesin game Unreal dan Unity menyediakan alat dan alur kerja yang merampingkan langkah Anda untuk memulai dan menjalankan game. Amazon GameLift Servers Amazon GameLift Serversadalah layanan hosting cloud yang dikelola sepenuhnya yang dapat digunakan pengembang game untuk mengelola dan menskalakan server game khusus untuk game multipemain berbasis sesi.

Versi plugin terbaru menawarkan penyempurnaan berikut:

- Alur kerja terpandu untuk hosting dengan Kontainer Terkelola. Alur kerja ini memandu Anda melalui langkah-langkah untuk menyiapkan gambar kontainer dengan perangkat lunak server game Anda, dan menerapkan solusi hosting berbasis cloud untuk server game Anda. Alur kerja menawarkan dua skenario penerapan yang berbeda: penerapan sederhana dan penerapan yang lebih lengkap dengan antrian penempatan sesi game dan mak comblang. FlexMatch Setiap skenario menghasilkan armada Amazon GameLift Servers kontainer dan AWS sumber daya pendukung.

- Proses yang ditingkatkan untuk menyiapkan profil AWS pengguna dan mengelola kredensial AWS akses untuk penggunaan plugin. Anda dapat mempertahankan beberapa profil agar berfungsi dengan berbagai AWS akun, pengguna akun, dan wilayah.
- Fungsionalitas tambahan untuk memperbarui armada kontainer yang ada. Anda dapat menerapkan gambar kontainer baru (seperti untuk pembaruan versi server game) dan mengubah pengaturan konfigurasi armada tanpa harus memulai dari awal.
- Alur kerja yang ditingkatkan untuk hosting dengan armada Amazon GameLift Servers Anywhere dan Managed. EC2 Perbaikan berdasarkan umpan balik pelanggan mencakup panduan yang lebih baik dengan tips dan tautan ke sumber daya yang bermanfaat.

Skenario penerapan untuk Kontainer Terkelola dan EC2 solusi Terkelola menggunakan AWS CloudFormation templat untuk membuat dan menyebarkan AWS sumber daya untuk setiap skenario. Template ini termasuk dalam download Amazon GameLift Servers plugin dan dapat diedit. Anda dapat menggunakannya apa adanya atau memodifikasinya untuk permainan Anda.

Pelajari lebih lanjut:

- [Plugin untuk Unreal: Menyebar game Anda ke armada kontainer terkelola, Panduan Pengembang Amazon GameLift Servers](#)
- [Plugin for Unity: Menyebar game Anda ke armada kontainer terkelola](#), Panduan Amazon GameLift ServersPengembang
- [Unduh plugin untuk Unreal dari GitHub](#)
- [Unduh plugin untuk Unity dari GitHub](#)
- [Hosting game dengan Amazon GameLift Servers](#)
- [Amazon GameLift Serversforum](#)

12 November 2024: Amazon GameLift Servers meluncurkan hosting game multipemain dengan Managed Containers

Versi SDK yang diperbarui:

- AWS SDK 1.11.445
- Server SDK, versi 5.2.0 (semua bahasa)

Amazon GameLift Servers rilis untuk ketersediaan umum solusi hosting baru untuk beban kerja server game dalam peti kemas. Dengan rilis ini, pengembang game sekarang dapat memanfaatkan manfaat kontainerisasi termasuk lingkungan yang konsisten dan aman, proses penyebaran yang disederhanakan, dan pemanfaatan sumber daya yang dioptimalkan.

Armada kontainer terkelola menggunakan EC2 instans Amazon yang dikelola oleh Amazon GameLift Servers atas nama Anda dan berdasarkan konfigurasi Anda. Anda membangun arsitektur kontainer khusus untuk game Anda dan menyediakan gambar kontainer dengan menyimpannya di repositori Amazon Elastic Container Registry (Amazon ECR). Armada kontainer hanya tersedia untuk server game berbasis Linux. Server game harus terintegrasi dengan Server SDK 5.2.0 atau lebih tinggi.

Dengan armada kontainer terkelola, Anda mendapatkan manfaat yang sama seperti EC2 armada yang dikelola. Ini termasuk dukungan untuk jenis instans On-Demand dan Spot, penskalaan kapasitas cerdas, penempatan sesi game dengan antrian, dan perjodohan. Anda juga mendapatkan metrik yang sama dengan jenis armada lainnya bersama dengan beberapa yang baru untuk kontainer. Fitur lain untuk armada kontainer meliputi:

- Penyelarasan dengan pengalaman tanpa server untuk beban kerja kontainer. Jalankan satu proses server game per kontainer dan kemas banyak kontainer ke setiap instance armada untuk penggunaan sumber daya yang optimal. Jika Anda lebih suka memiliki wadah dengan beberapa proses server game, Anda dapat menggunakan Amazon GameLift Servers Agen untuk manajemen host otomatis.
- Pembuatan armada yang efisien. Armada kontainer dirancang untuk memerlukan pengaturan konfigurasi penerapan minimal, dengan nilai yang masuk akal suggested/default . Anda dapat dengan cepat menyebarkan armada kerja, dan kemudian menyesuaikan pengaturan individual sesuai kebutuhan.
- Alat pembuatan versi untuk arsitektur kontainer. Sekarang Anda dapat memperbarui definisi grup kontainer (yang mirip dengan “tugas” penampung), memelihara beberapa versi, dan menentukan versi mana yang akan diterapkan ke armada.
- Alat pembaruan armada. Dengan armada kontainer, Anda tidak perlu lagi membuat armada baru saat ingin merilis pembaruan versi server game. Sebagai gantinya, Anda sekarang dapat memperbarui gambar kontainer Anda dan menerapkan pembaruan ke armada yang ada.

Anda dapat membangun armada Amazon GameLift Servers kontainer di Wilayah AWS mana saja yang Amazon GameLift Servers mendukung armada multi-lokasi, dan Anda dapat menyebarkan instance armada kontainer ke lokasi terpencil yang didukung. Untuk detail selengkapnya, lihat

[Amazon GameLift Servers lokasi layanan](#). Kontainer terkelola saat ini tidak tersedia di Wilayah AWS Tiongkok.

Pelajari lebih lanjut:

- Posting blog: [Manfaatkan kontainer yang dikelola sepenuhnya untuk menyelenggarakan game multipemain dalam skala global Amazon GameLift Servers](#)
- [Kontainer yang dikelola](#) ikhtisar, Panduan Amazon GameLift Servers Pengembang
- [Cara kerja kontainer Amazon GameLift Servers](#), Panduan Amazon GameLift Servers Pengembang
- [Peta jalan pengembangan untuk hosting dengan kontainer Amazon GameLift Servers terkelola](#), Panduan Amazon GameLift Servers Pengembang
- [CreateContainerFleet](#), Referensi Amazon GameLift Servers API

19 September 2024: Amazon GameLift Servers merilis pembaruan ke SDK server C++ dan plugin untuk Unreal Engine

Versi SDK yang diperbarui:

C++ Server SDK, versi 5.1.3

- Kemampuan logging baru. Anda sekarang dapat mengakses log permintaan SDK.
- Peningkatan keandalan transmisi pesan SDK. SDK sekarang menggunakan mekanisme rekoneksi yang lebih kuat untuk memulihkan jika terjadi gangguan jaringan atau penurunan pesan acak.

Diperbarui versi plugin:

Amazon GameLift Servers plugin untuk Unreal Engine, versi 1.1.2

- Upgrade untuk mendukung versi terbaru dari server C++ SDK 5.1.3.
- Di Amazon GameLift Servers plugin Unreal Engine, saat menjelajah server build yang dapat dieksekusi untuk armada, Anda sekarang memiliki opsi untuk menelusuri Semua File.

Plugin SDK Server C++ untuk Unreal, versi 5.1.2

- Upgrade untuk mendukung versi terbaru dari server C++ SDK 5.1.3.

Pelajari lebih lanjut:

- [Mengintegrasikan game dengan Amazon GameLift Servers plugin untuk Unreal Engine](#), Amazon GameLift Servers Panduan Pengembang
- [Amazon GameLift Servers unduhan plugin dan SDK](#)

5 September 2024: Amazon GameLift Servers meningkatkan observabilitas proses pembuatan armada

Berdasarkan umpan balik pelanggan, kami telah mengklarifikasi Amazon GameLift Servers alur kerja untuk membuat EC2 armada terkelola dan menyiapkannya untuk menyelenggarakan sesi permainan. Peningkatan meliputi:

- Kami telah memberikan deskripsi yang lebih spesifik dan akurat dari setiap fase proses pembuatan armada. Visibilitas yang ditingkatkan ini memudahkan untuk menentukan dan menyelesaikan masalah lebih cepat.
- Fase Membangun dan Mengaktifkan lebih baik memisahkan tugas penyebaran instance (membangun) dari tugas untuk memulai proses server game dan terhubung ke Amazon GameLift Servers layanan (mengaktifkan). Perubahan ini membuatnya lebih mudah untuk mengenali kemungkinan penyebab masalah. Selain itu, Anda sekarang dapat terhubung dari jarak jauh ke armada saat mereka berada dalam fase Aktivasi.
- Dua acara pembuatan armada baru mengkomunikasikan keberhasilan atau kegagalan skrip pemasangan server game. Jika Anda membangun server game menyertakan skrip penginstalan, Amazon GameLift Servers mencoba menjalankan skrip dan memancarkan salah satu peristiwa baru berikut:
 - FLEET_CREATION_COMPLETED_INSTALLER
 - FLEET_CREATION_FAILED_INSTALLER

Pelajari lebih lanjut:

- [Cara kerja pembuatan Amazon GameLift Servers armada](#), Panduan Amazon GameLift Servers Pengembang
- [Masalah Amazon GameLift Servers armada debug](#), Panduan Amazon GameLift Servers Pengembang
- [Tipe data peristiwa](#), Referensi Amazon GameLift Servers API

15 Agustus 2024: Amazon GameLift Servers meningkatkan pengalaman konsol

Berdasarkan umpan balik pelanggan, kami telah membuat pembaruan berikut untuk pengalaman [Amazon GameLift Servers konsol](#):

- Preferensi tampilan Anda untuk halaman sekarang secara otomatis disimpan ke pengguna AWS akun Anda dan diterapkan setiap kali Anda kembali ke halaman. Preferensi tampilan memungkinkan Anda memilih informasi apa yang akan disertakan dalam tampilan tabel, seperti pada halaman daftar Armada. Sesuaikan preferensi tampilan Anda dengan menggunakan  ikon di sudut kanan atas tabel.
- Alur kerja Buat Armada untuk armada EC2 terkelola telah disederhanakan untuk menggabungkan pemilihan lokasi armada dan jenis instance. Kami telah mempermudah Anda menemukan jenis instans yang tepat untuk armada Anda, bahkan ketika Anda mengubah pilihan lokasi.

Pelajari lebih lanjut:

- [Buat EC2 armada Amazon GameLift Servers terkelola](#), Panduan Amazon GameLift Servers Pengembang

25 Juli 2024: Amazon GameLift Servers menambahkan dukungan untuk AWS Zona Lokal Nigeria

Dengan hosting Amazon GameLift Servers terkelola, Anda sekarang dapat menyebarkan sumber daya server game di Nigeria, Afrika Barat, dan memperluas jangkauan game Anda ke pemain di seluruh Afrika. Gunakan AWS Local Zones untuk menempatkan server game secara geografis lebih dekat dengan pemain Anda untuk mengurangi latensi dan meningkatkan gameplay secara signifikan.

Untuk segera mulai menyelenggarakan sesi permainan di Nigeria, tambahkan Zona Lokal Nigeria baru sebagai lokasi terpencil ke armada multi-lokasi baru atau yang sudah ada. Jika game Anda menggunakan Amazon GameLift ServersFlexMatch, perbarui armada dalam antrian perjodohan Anda untuk menyertakan Zona Lokal baru. Dengan armada multi-lokasi, Anda dapat langsung mengelola kapasitas hosting di setiap lokasi.

Induk Wilayah AWS untuk Lagos, Zona Lokal Nigeria adalah Wilayah Afrika (Cape Town) af-south-1 (), Amazon GameLift Servers yang juga mendukung sebagai lokasi terpencil. Nama Zona Lokal Nigeria adalah. af-south-1-los-1

Pelajari lebih lanjut:

- [Amazon GameLift Servers lokasi layanan](#), Panduan Amazon GameLift Servers Pengembang
- [Perbarui lokasi armada](#), Panduan Amazon GameLift Servers Pengembang

02 Juli 2024: Amazon GameLift Servers merilis alat konsol baru untuk melihat data sesi pemain

Amazon GameLift Servers Konsol sekarang menawarkan alat pencarian sesi pemain yang memungkinkan Anda mengambil informasi sesi pemain dengan ID sesi game, ID sesi pemain, atau ID pemain. Game yang menggunakan FlexMatch perjodohan secara otomatis menghasilkan sesi pemain untuk setiap pemain yang cocok. Untuk semua game lainnya, sesi pemain adalah fitur opsional.

Anda dapat menemukan alat pencarian sesi pemain di navigasi utama untuk Amazon GameLift Servers konsol. Lihat sesi pemain individual atau bandingkan data di beberapa sesi pemain. Anda juga dapat membuka data sesi pemain saat melihat halaman detail sesi permainan.

Pelajari lebih lanjut:

- [Sesi permainan dan pemain di Amazon GameLift Servers konsol](#), Panduan Amazon GameLift Servers Pengembang

24 April 2024: Amazon GameLift Servers rilis pratinjau armada kontainer

Amazon GameLift Servers sekarang menawarkan pratinjau armada kontainer, yang memberi Anda peningkatan portabilitas, skalabilitas, toleransi kesalahan, dan kelincahan.

Dalam armada kontainer, EC2 instans Amazon menampung satu atau lebih kontainer Anda. Wadah ini termasuk server game Anda bersama dengan apa pun yang dibutuhkannya, termasuk dependensi dan konfigurasi. Contoh dependensi termasuk SDKs dan paket perangkat lunak. Setelah Anda mengunggah kontainer Anda ke Amazon Elastic Container Registry pribadi Anda, Amazon GameLift Servers isi armada Anda dengan kontainer.

Agar berfungsi dalam armada kontainer, server game Anda harus berjalan di Linux dan terintegrasi dengan Server SDK 5.x. Dalam armada kontainer, Anda memiliki kontrol sumber daya hosting yang disesuaikan sehingga Anda dapat mengoptimalkan konsumsi sumber daya seperti unit CPU dan memori. Anda juga dapat meng-host beberapa server game dalam wadah untuk mengurangi penggunaan sumber daya.

Dalam armada kontainer, Anda mendapatkan banyak manfaat yang sama dengan jenis armada lain seperti jenis instans On-Demand, penskalaan (otomatis dan manual), antrian, dan perijodohan. Anda juga mendapatkan metrik yang sama dengan jenis armada lainnya bersama dengan beberapa yang baru untuk kontainer. Armada kontainer memberi Anda jangkauan global ke pemain di wilayah lokasi ini:

- ap-northeast-1
- ap-northeast-2
- ap-southeast-2
- eu-central-1
- eu-west-1
- us-east-1
- us-west-2

Untuk menjangkau lebih banyak wilayah dan zona lokal, buat armada kontainer multi-lokasi.

Pelajari lebih lanjut:

- [Mengelola hosting dengan Amazon GameLift Servers kontainer](#), Panduan Amazon GameLift Servers Pengembang
- [CreateContainerGroupDefinition](#), Referensi Amazon GameLift Servers API

13 Februari 2024: Amazon GameLift Servers meluncurkan peningkatan SDKs, dan menyederhanakan pemasangan Amazon GameLift Servers plugin untuk Unreal Engine

Versi SDK yang diperbarui:

- Go Server SDK, versi 5.1.0
- C# Server SDK, versi 5.1.2
- C++ Server SDK, versi 5.1.2

Kami melakukan perbaikan berikut:

- Meningkatkan keandalan SDK dengan menambahkan rekoneksi otomatis jika terjadi gangguan jaringan.
- [Go] Sekarang Anda dapat menelepon `InitSDK()` dengan atau tanpa parameter server. Server game yang berjalan pada EC2 armada Amazon GameLift Servers terkelola membaca parameter server langsung dari variabel lingkungan. Server game di armada Amazon GameLift Servers Anywhere harus menelepon `InitSDK()` dengan parameter server.

Diperbarui versi plugin:

- Amazon GameLift Serversplugin untuk Unreal Engine, versi 1.1.0
- Amazon GameLift Serversplugin untuk Unity, versi 2.1.0
- Plugin SDK Server C ++ untuk Unreal, versi 5.1.1
- Plugin SDK Server C # untuk Unity, versi 5.1.2

Kami melakukan perbaikan berikut:

- [Amazon GameLift ServersPlugin untuk Unreal Engine] Memperbarui instruksi instalasi dan menyederhanakan kemasannya. Plugin ini sekarang menyertakan versi terbaru dari C++ Server SDK untuk Unreal.
- Memutakhirkan plugin untuk mendukung versi terbaru SDK server untuk. Amazon GameLift Servers

Pelajari lebih lanjut:

- [Mengintegrasikan game dengan Amazon GameLift Servers plugin untuk Unreal Engine](#), Amazon GameLift Servers Panduan Pengembang
- [Amazon GameLift Serversunduhan plugin dan SDK](#)

14 Desember 2023: Amazon GameLift Servers menambahkan kemampuan untuk memperbarui properti game dari sesi game aktif

Anda sudah dapat mengatur properti game saat membuat sesi game, dan untuk mencari sesi game untuk properti tertentu. Sekarang Anda juga dapat menambahkan dan memperbarui properti ini dalam sesi permainan aktif.

Misalnya, pemain Anda memilih pada peta yang ingin mereka mainkan. Klien game Anda memanggil `UpdateGameSession` untuk memodifikasi `GameProperty` nilai `{"Key": "map", "Value": "jungle"}`. Game Anda kemudian mengimplementasikan peta baru untuk para pemain di sesi permainan.

Administrator game juga dapat mengambil data yang berguna dari properti game dengan menggunakan operasi `SearchGameSessions`. Misalnya, administrator dapat membuat daftar sesi permainan yang memiliki `Status` nilai `ACTIVE` dan properti game ini: `{"Key": "map", "Value": "desert"}`.

Pelajari lebih lanjut:

- [the section called “Tambahkan Amazon GameLift Servers ke klien game”](#), Panduan Amazon GameLift Servers Pengembang
- [GameProperty](#), Referensi Amazon GameLift Servers API
- [UpdateGameSession](#), Referensi Amazon GameLift Servers API
- [SearchGameSessions](#), Referensi Amazon GameLift Servers API

21 November 2023: Amazon GameLift Servers meluncurkan dukungan untuk Infrastruktur sebagai alat Kode seperti Terraform dan Pulumi didukung oleh AWS Cloud Control API

Anda sekarang dapat mengelola seluruh tumpukan Amazon GameLift Servers sumber daya Anda menggunakan alat Infrastructure as Code (IaC). Alat-alat ini termasuk AWS CloudFormation, dan juga alat pihak ketiga seperti Terraform dan Pulumi. Dengan dukungan tambahan ini, Anda sekarang dapat fokus untuk membangun game Anda, dan memanfaatkan DevOps strategi untuk menangani manajemen sumber daya, CI/CD, dan penyebaran ke pelanggan Anda.

Sekarang Anda juga dapat menyediakan dan mengonfigurasi semua jenis Amazon GameLift Servers sumber daya dengan menggunakan AWS Cloud Control API. Anda dapat terus bekerja dengan sumber daya menggunakan Amazon GameLift Servers APIs atau AWS CloudFormation templat untuk Amazon GameLift Servers.

Untuk detail tentang Amazon GameLift Servers sumber daya yang tersedia melalui IaC, lihat [referensi tipe Amazon GameLift Servers sumber daya referensi tipe](#) Amazon GameLift Servers sumber daya.

Selain itu, Anda sekarang dapat secara otomatis menskalakan armada Anda menggunakan AWS CloudFormation template atau AWS Cloud Control API dengan menggunakan properti [Fleet](#) baru: `ScalingPolicies`.

Cloud Control API memberi pengembang seperangkat standar APIs untuk membuat, membaca, memperbarui, menghapus, dan mencantumkan sumber daya (CRUDL) di ratusan AWS layanan dan beberapa alat pihak ketiga seperti Terraform dan Pulumi.

Pelajari lebih lanjut:

- [AWS CloudFormation](#)
- [AWS API Kontrol Awan](#)
- [AWS Penyedia CC Terraform](#)
- [Pulumi](#)

16 November 2023: Amazon GameLift Servers memperbarui plugin mandiri untuk Unity

Versi SDK yang diperbarui: Amazon GameLift Servers plugin untuk Unity, versi 2.0.0

Amazon GameLift ServersPlugin untuk Unity menyediakan alat dan alur kerja yang merampingkan langkah-langkah untuk membuat game Unity Anda aktif dan berjalan untuk hosting cloud. Amazon GameLift Servers Amazon GameLift Servers adalah layanan yang dikelola sepenuhnya yang memungkinkan pengembang game mengelola dan menskalakan server game khusus untuk game multipemain berbasis sesi.

Dengan versi ini, plugin untuk Unity diperbarui untuk menggunakan Amazon GameLift Servers fitur terbaru, termasuk server SDK versi 5.x dan dukungan untuk pengujian lokal dengan Amazon GameLift Servers Anywhere. Plugin ini kompatibel dengan Unity versi Unity 2021.3 LTS dan 2022.3 LTS.

Fitur plugin utama meliputi:

- Alur kerja UI terpandu di editor Unity untuk skenario berikut:
 - Uji integrasi game Anda dengan Amazon GameLift Servers menggunakan workstation lokal Anda sebagai host. Alur kerja ini membantu Anda menyiapkan armada Amazon GameLift Servers Anywhere untuk mesin lokal Anda, meluncurkan instance server dan klien game Anda, meminta sesi game Amazon GameLift Servers, dan bergabung dengan game.

- Terapkan solusi cloud hosting untuk server game terintegrasi Anda dengan AWS sumber daya Amazon GameLift Servers terkelola EC2 dan pendukung. Alur kerja ini membantu Anda mengonfigurasi game untuk cloud hosting, dan menyediakan tiga skenario penerapan:
 - Menyebarkan server game ke satu armada.
 - Terapkan server game ke satu set armada Spot berbiaya rendah di beberapa Wilayah. AWS
 - Menyebarkan server game dengan FlexMatch mak comblang.
- Kemampuan untuk mengatur profil pengguna yang menautkan ke pengguna AWS akun dan menetapkan AWS Wilayah default. Anda dapat mempertahankan beberapa profil agar berfungsi di berbagai AWS akun, pengguna akun, dan wilayah.
- Kemudahan khusus yang membantu merampingkan proses Amazon GameLift Servers integrasi dan penyebaran, termasuk:
 - Setiap solusi hosting mencakup AWS sumber daya pendukung, termasuk kumpulan pengguna Amazon Cognito yang menyediakan validasi pemain IDs dan pemain unik. Solusinya juga mencakup bucket Amazon S3 untuk penyimpanan, notifikasi acara Amazon SNS, fungsi AWS Lambda , dan sumber daya lainnya.
 - Untuk alur kerja Anywhere, plugin mengotomatiskan pengaturan parameter server yang diperlukan.
 - Untuk EC2 alur kerja Amazon, setiap solusi penerapan menyediakan layanan backend klien bawaan menggunakan fungsi Lambda. Layanan backend berada di antara klien game dan Amazon GameLift Servers layanan dan mengelola semua panggilan langsung ke layanan. Amazon GameLift Servers
- Konten untuk pengujian integrasi, termasuk aset dan kode untuk contoh game multiplayer sederhana untuk menggambarkan server game dan integrasi klien game.
- Dokumentasi plugin dengan panduan integrasi terperinci dan kode sampel.

Semua skenario penerapan, termasuk untuk EC2 armada Anywhere dan Amazon, menggunakan AWS CloudFormation template untuk mendeskripsikan dan menyebarkan AWS sumber daya untuk solusi game Anda. Template ini termasuk dalam download Amazon GameLift Servers plugin. Anda dapat menggunakannya apa adanya atau menyesuaikannya untuk game Anda.

Pelajari lebih lanjut:

- [Amazon GameLift Serversplugin untuk Unity \(server SDK 5.x\)](#), Panduan Amazon GameLift Servers Pengembang
- [Unduh plugin dari GitHub](#)

- [Tentang Amazon GameLift Servers hosting](#)
- [Amazon GameLift Serversforum](#)

2 November 2023: Amazon GameLift Servers menambahkan dukungan untuk kredensi bersama

Versi SDK yang diperbarui: AWS SDK 1.11.193

Fitur Amazon GameLift Servers kredensi bersama yang baru memungkinkan aplikasi yang digunakan pada EC2 armada terkelola untuk berinteraksi dengan sumber daya lain. AWS Pembaruan ini memengaruhi aplikasi yang Anda bundel dan terapkan bersama dengan binari server game yang terintegrasi dengan server SDK versi 5.x atau yang lebih baru. (Executable server game sudah dapat meminta kredensial menggunakan aksi SDK 5.x server.) `GetFleetRoleCredentials()`

Misalnya, jika Anda ingin menerapkan build server game dengan CloudWatch agen Amazon untuk mengumpulkan metrik EC2 instans dan data lainnya, agen memerlukan izin untuk berinteraksi dengan sumber daya Anda CloudWatch . Untuk melakukannya, Anda harus terlebih dahulu menyiapkan peran AWS Identity and Access Management (IAM) dengan izin untuk menggunakan CloudWatch sumber daya, lalu mengonfigurasi armada dengan peran IAM dan kredensial bersama diaktifkan. Saat Amazon GameLift Servers menerapkan build server game Anda ke setiap EC2 instance, itu menghasilkan file kredensi bersama dan menyimpannya di instance. Semua aplikasi pada instance dapat menggunakan kredensi bersama. Amazon GameLift Servers secara otomatis menyegarkan kredensial sementara sepanjang masa pakai instance.

Anda dapat mengaktifkan kredensial bersama saat membuat EC2 armada terkelola menggunakan metode berikut:

- Dalam alur kerja pembuatan armada Amazon GameLift Servers konsol.
- Saat memanggil operasi API layanan `CreateFleet` menggunakan parameter `newInstanceRoleCredentialsProvider`.
- Saat memanggil operasi AWS CLI `aws gamelift create-fleet` dengan parameter `instance-role-credentials-provider`

Pelajari lebih lanjut:

- [Berkomunikasi dengan AWS sumber daya lain dari armada Anda](#), Panduan Amazon GameLift ServersPengembang

- [CreateFleet, InstanceRoleCredentialsProvider](#), Referensi Amazon GameLift Servers API
- [Siapkan peran layanan IAM, Panduan Amazon GameLift Servers](#) Pengembang

28 September 2023: Amazon GameLift Servers merilis plugin mandiri baru untuk Unreal Engine Engine

Versi SDK yang diperbarui: Amazon GameLift Servers plugin untuk Unreal Engine versi 1.0.0

Amazon GameLift ServersPlugin untuk Unreal Engine menyediakan alat dan alur kerja yang merampingkan langkah-langkah Anda untuk memulai dan menjalankan game Amazon GameLift Servers untuk hosting cloud. Amazon GameLift Serversadalah layanan yang dikelola sepenuhnya yang memungkinkan pengembang game mengelola dan menskalakan server game khusus untuk game multipemain berbasis sesi. Plugin ini mendukung UE versi 5.0, 5.1, dan 5.2. Fitur utama meliputi:

- Alur kerja UI terpandu di editor Unreal] melewati jalur berikut:
 - Uji integrasi game Anda dengan Amazon GameLift Servers menggunakan workstation lokal Anda sebagai host. Alur kerja ini membantu Anda menyiapkan armada Amazon GameLift Servers Anywhere untuk mesin lokal Anda, meluncurkan instance server dan klien game Anda, meminta sesi gameAmazon GameLift Servers, dan mendapatkan informasi koneksi untuk sesi game baru.
 - Terapkan solusi hosting EC2 cloud Amazon untuk server game terintegrasi Anda. Alur kerja ini membantu Anda mengonfigurasi game Anda untuk cloud hosting, dan menyediakan tiga skenario penerapan yang berbeda: menyebarkan ke satu armada, menyebarkan ke satu set armada spot di beberapa wilayah, atau menyebarkan ke satu set armada dengan mak comblang. FlexMatch Solusi untuk setiap skenario penerapan mencakup Amazon GameLift Servers sumber daya dan sumber AWS daya pendukung.
- Kemampuan untuk mengatur profil pengguna yang menautkan ke pengguna AWS akun dan menentukan AWS Wilayah default. Anda dapat mempertahankan beberapa profil agar berfungsi di berbagai AWS akun, pengguna akun, dan wilayah.
- Kemudahan khusus yang membantu merampingkan proses Amazon GameLift Servers integrasi dan penyebaran, termasuk:
 - Setiap solusi hosting mencakup AWS sumber daya pendukung, termasuk kumpulan pengguna Amazon Cognito dasar yang menyediakan pemutar unik IDs, bucket Amazon S3 untuk penyimpanan, pemberitahuan acara Amazon SNS, dan fungsi. AWS Lambda

- Untuk alur kerja Anywhere, plugin mengotomatiskan pengaturan parameter server yang diperlukan menggunakan argumen baris perintah.
- Untuk EC2 alur kerja Amazon, setiap solusi penerapan menyediakan layanan backend klien bawaan menggunakan fungsi Lambda. Layanan backend menerima permintaan dari klien game dan meneruskannya ke layanan. Amazon GameLift Servers
- Konten untuk pengujian integrasi, termasuk peta permainan pemula dan dua peta pengujian dengan cetak biru dasar dan elemen UI.
- Dokumentasi plugin dengan panduan integrasi terperinci dan kode sampel.

Semua skenario penerapan, termasuk untuk EC2 armada Anywhere dan Amazon, menggunakan AWS CloudFormation template untuk menjelaskan solusinya. Plugin menggunakan template ini saat menyebarkan Amazon GameLift Servers sumber daya untuk game Anda. Template ini termasuk dalam download Amazon GameLift Servers plugin dan dapat diedit. Anda dapat menggunakannya apa adanya atau memodifikasinya untuk permainan Anda.

Pelajari lebih lanjut:

- [Amazon GameLift Serversplugin untuk Unreal Engine](#), Panduan Amazon GameLift Servers Pengembang
- [Unduh plugin dari GitHub](#)
- [Tentang Amazon GameLift Servers hosting](#)
- [Amazon GameLift Serversforum](#)

17 Agustus 2023: Amazon GameLift Servers menawarkan hosting server game dengan prosesor AWS Graviton

Versi SDK yang diperbarui: AWS SDK 1.11.144

Dengan Amazon GameLift Servers Anda sekarang dapat meng-host game Anda di cloud menggunakan EC2 instance dengan prosesor AWS Graviton. Dirancang oleh AWS prosesor berbasis ARM64, instans Graviton memberikan kinerja harga terbaik untuk penggunaan beban kerja cloud EC2, dengan peningkatan hingga 40% dibandingkan instans berbasis x86 yang sebanding. Prosesor Graviton3 terbaru menawarkan kinerja komputasi hingga 25% lebih baik dibandingkan versi sebelumnya.

Dengan Amazon GameLift Servers, Anda sekarang dapat memilih dari instance baru ini dalam keluarga AWS Graviton:

- Contoh berbasis Graviton2: c6g, c6gn, r6g, m6g, g5g
- Contoh berbasis Graviton3: c7g, r7g, m7g

Pelajari lebih lanjut:

- [AWS Prosesor Graviton](#): Pelajari tentang manfaat dan penggunaan praktis instance berbasis EC2 Graviton.
- [Memulai Graviton](#): Dapatkan ikhtisar tentang instans berbasis Graviton dan wawasan tentang bagaimana aplikasi berjalan pada mereka tergantung pada sistem operasi, bahasa, dan waktu berjalan mereka.

 Note

Instans Graviton Arm memerlukan Amazon GameLift Servers server yang dibangun di OS Linux. Server SDK 5.1.1 atau yang lebih baru diperlukan untuk C++ dan C#. Server SDK 5.0 atau yang lebih baru diperlukan untuk Go. Instans ini tidak memberikan out-of-the-box dukungan untuk instalasi Mono di Amazon Linux 2023 (AL2023) atau Amazon Linux 2 (). AL2

27 Juli 2023: Amazon GameLift Servers merilis server SDK 5.1.0 dengan dukungan tambahan untuk pengembangan Unity

Versi SDK yang diperbarui: Server SDK untuk C++, C#/Unity, Unreal 5.1.0

Rilis terbaru SDK Amazon GameLift Servers server memberikan pembaruan untuk C++, C#, dan plugin Unreal, dan plugin baru untuk digunakan dengan mesin game Unity. Pengembang game mengintegrasikan SDK Amazon GameLift Servers server ke server game yang mereka gunakan untuk hosting. Amazon GameLift Servers

Versi SDK server terbaru berisi pembaruan berikut, yang mencakup sejumlah permintaan pelanggan:

- Unduh paket SDK khusus bahasa — [Situs Amazon GameLift Servers unduhan](#) yang diperbarui berisi paket SDK untuk setiap bahasa. Anda dapat mengunduh versi saat ini atau sebelumnya.

- Plugin SDK server C # baru untuk Unity - [Paket SDK server baru untuk Unity berisi pustaka C# bawaan yang dapat Anda instal menggunakan manajer paket di Unity Editor \(lihat panduan integrasi Unity yang baru\)](#). Pustaka ini menyertakan dependensi yang diperlukan melalui UnityNuGet Anda dapat menggunakan plugin ini dengan Unity 2020.3 LTS, 2021.3 LTS dan 2022.3 LTS untuk Windows dan Mac OS. Ini mendukung profil .NET Framework dan .NET Standard Unity, dengan .NET Standard 2.1 dan .NET 4.x.
- Solusi .NET terkonsolidasi untuk C # - Server SDK untuk C# sekarang mendukung .NET Framework 4.6.2 (ditingkatkan dari 4.6.1) dan .NET 6.0 dalam satu solusi. .NET Standard 2.1 tersedia dengan pustaka Unity-built.
- Pembaruan Server SDK 5.1.0
 - [C ++, C #, Unreal] Anda sekarang dapat memanggil `InitSDK()` dengan atau tanpa parameter server. Server game yang berjalan pada EC2 armada Amazon GameLift Servers terkelola membaca parameter server langsung dari variabel lingkungan. Server game di armada Amazon GameLift Servers Anywhere harus menelepon `InitSDK()` dengan parameter server.
 - [C ++, C #, Unreal] Panggilan SDK server telah meningkatkan pesan kesalahan.
 - [C++ SDK] Untuk meningkatkan waktu pembuatan SDK Server, flag build dinonaktifkan `-DRUN_CLANG_FORMAT` secara default. Anda dapat mengaktifkannya dengan `-DRUN_CLANG_FORMAT=1`.
 - [C++ SDK] Saat membangun pustaka tanpa pustaka standar (`-DGAMELIFT_USE_STD=0`), `InitSDK()` tidak lagi menggunakan tipe data. `std::`
- Dokumentasi SDK 5.x server yang diperluas
 - Panduan referensi SDK server yang diperbarui untuk C ++, C #/Unity, dan Unreal termasuk cakupan yang diperluas dari semua tipe data.
 - [C# server SDK 5.x untuk Amazon GameLift Servers -- Tindakan](#)
 - [Server C ++ SDK 5.x untuk Amazon GameLift Servers -- Tindakan](#)
 - [Server C ++ \(Tidak nyata\) SDK 5.x untuk -- Tindakan Amazon GameLift Servers](#)
 - Versi baru dari panduan integrasi SDK 5 server untuk plugin Unity dan Unreal
 - [Integrasikan Amazon GameLift Servers ke dalam proyek Unity](#)
 - [Integrasikan Amazon GameLift Servers ke dalam proyek Unreal Engine](#)
- Pembaruan dokumentasi tambahan
 - Dokumentasi yang direvisi untuk operasi API Amazon GameLift Servers layanan [GetComputeAccess](#) dan [GetInstanceAccess](#) untuk memperjelas prosedur akses jarak jauh berdasarkan versi SDK Amazon GameLift Servers server yang digunakan.

- Deskripsi yang direvisi [GameSessionPlacement](#) untuk mendokumentasikan bagaimana informasi sesi game bersifat sementara ketika penempatan berada dalam status “tertunda”.

13 Juli 2023: Amazon GameLift Servers menambahkan metrik perangkat keras armada

Sekarang Anda dapat melacak metrik kinerja perangkat keras untuk EC2 armada Amazon GameLift Servers terkelola Anda. Metrik mencakup metrik EC2 instance untuk pemanfaatan CPU, volume lalu lintas jaringan, dan aktivitas disk. read/write Untuk Amazon GameLift Servers, metrik ini menjelaskan semua instance aktif di lokasi armada. Anda dapat melihat metrik perangkat keras armada ini menggunakan CloudWatch dasbor Amazon di. AWS Management Console Anda juga dapat melihatnya di Amazon GameLift Servers konsol dalam detail armada.

Pelajari selengkapnya:

- [Monitor Amazon GameLift Servers dengan Amazon CloudWatch](#) (Metrik untuk armada), Panduan Pengembang Amazon GameLift Servers

29 Juni 2023: Amazon GameLift Servers meluncurkan dukungan untuk Amazon Linux 2023

Versi SDK yang diperbarui: AWS SDK 1.11.111

Amazon GameLift Servers pelanggan sekarang dapat menggunakan sistem operasi Amazon Linux 2023 untuk meng-host server game mereka. AL2023 menawarkan beberapa perbaikan AL2 termasuk keamanan. Sistem operasi ini tersedia di semua Wilayah AWS kecuali Wilayah Tiongkok.

Pelanggan dapat menggunakan sistem operasi Linux yang lebih baru dan terus menerima pembaruan keamanan penting ketika dukungan berakhir untuk Amazon Linux (AL1) pada Desember 2023. Support untuk Amazon Linux 2 berlanjut hingga 30 Juni 2025.

Pelajari selengkapnya:

- [Amazon GameLift Servers Tanya Jawab Server Linux](#)
- [Membandingkan Amazon Linux 2 dan Amazon Linux 2023](#)
- Amazon GameLift Servers Tautan Referensi API:
 - [AWS Tindakan SDK CreateBuild](#)

- [Perintah CLI `upload-build`](#)
- [Perintah CLI `create-build`](#)

25 Mei 2023: Amazon GameLift Servers FleetIQ menambahkan filter untuk mengecualikan penempatan sesi game pada instance yang menguras

Versi SDK yang diperbarui: AWS SDK 1.11.87

Jika Anda menggunakan Amazon GameLift Servers FleetIQ untuk hosting game, Anda sekarang dapat mencegah penempatan sesi game pada instance yang saat ini menguras tenaga. Instans yang menguras ditandai untuk shutdown, tetapi mereka masih dapat dipilih untuk menyelenggarakan sesi game baru jika tidak ada sumber daya hosting lain yang tersedia. Dengan fitur baru ini, Anda dapat mengecualikan penggunaan instance yang menguras seluruhnya.

Gunakan fitur ini saat menelepon `ClaimGameServer` untuk menemukan server game yang tersedia. Tambahkan `FilterOption` parameter baru dan atur status instance yang diizinkan ke `ACTIVE` saja. Sebagai tanggapan, hanya Amazon GameLift Servers FleetIQ melihat contoh aktif saat mencari dan mengklaim server game yang tersedia.

Pelajari lebih lanjut:

- [ClaimGameServer](#) di Referensi API Amazon GameLift Servers
- [Cara FleetIQ kerja](#) di Panduan Amazon GameLift Servers FleetIQ Pengembang

16 Mei 2023: Amazon GameLift Servers mendukung penandaan alokasi biaya untuk armada

Amazon GameLift Servers pelanggan sekarang dapat menggunakan tag alokasi AWS Billing biaya untuk mengatur biaya hosting game mereka. Anda dapat menetapkan tag alokasi biaya ke sumber daya Amazon GameLift Servers EC2 armada individu untuk melacak bagaimana armada Anda berkontribusi terhadap biaya hosting secara keseluruhan.

Pelajari lebih lanjut:

- [Kelola Anda Amazon GameLift Servers biaya hosting](#)
- [Menggunakan tag alokasi AWS biaya](#), AWS Billing Panduan Pengguna

20 April 2023: Amazon GameLift Servers meluncurkan dukungan untuk Windows Server 2016

Versi SDK yang diperbarui: AWS SDK 1.11.63

Amazon GameLift ServersPelanggan sekarang dapat menggunakan sistem operasi Windows Server 2016 untuk meng-host server game mereka. Sistem operasi ini tersedia di semua Wilayah AWS. Pelanggan dapat menggunakan sistem operasi Windows yang lebih baru dan terus menerima pembaruan keamanan penting karena Microsoft mengakhiri dukungannya untuk Windows Server 2012 pada Oktober 2023.

Mulai hari ini, pelanggan baru yang membutuhkan lingkungan runtime Windows harus menentukan Windows Server 2016 saat membuat build server game baru untuk hosting. Pelanggan yang sudah ada dapat terus membuat build dan armada baru dengan Windows Server 2012 tetapi harus menyelesaikan migrasi dengan Windows Server 2016 sebelum tanggal dukungan Microsoft berakhir pada 10 Oktober 2023.

Pembaruan ini mencakup perubahan layanan berikut:

- Saat membuat build server game menggunakan perintah Amazon GameLift Servers SDK atau CLI, Anda sekarang harus secara eksplisit mengatur sistem operasi. Tidak ada lagi nilai default. Untuk menyebarkan server game Anda di Windows Server 2016, gunakan nilainya `WINDOWS_2016`.
- Saat membuat build server game menggunakan Amazon GameLift Servers konsol, Anda harus memilih sistem operasi dari nilai yang tersedia. Jika Anda adalah pelanggan lama dengan armada Windows Server 2012 aktif, Anda dapat memilih salah satu `WINDOWS_2012` atau `WINDOWS_2016`.

Pelajari lebih lanjut:

- Amazon GameLift ServersTautan Referensi API:
 - [Perintah CLI `upload-build`](#)
 - [Perintah CLI `create-build`](#)
 - [AWS Tindakan SDK `CreateBuild`](#)
- [Amazon GameLift ServersFAQ untuk Windows 2012](#)

13 April 2023: Amazon GameLift Servers meluncurkan server SDK 5.x untuk Unreal

Versi SDK yang diperbarui: Server SDK 5.0.0 untuk Unreal

Versi terbaru dari plugin Amazon GameLift Servers ringan untuk Unreal Engine sekarang didasarkan pada Amazon GameLift Servers server SDK 5.x. Untuk mulai mengintegrasikan lingkungan Unreal Engine Anda dengan Amazon GameLift Servers lihat tautan berikut.

Pelajari lebih lanjut:

- [Integrasikan Amazon GameLift Servers ke dalam proyek Unreal Engine](#)
- [Tambahkan Amazon GameLift Servers ke server game Anda](#)
- [Server C ++ SDK 5.x untuk Amazon GameLift Servers -- Tindakan](#)

14 Maret 2023: Amazon GameLift Servers meluncurkan pengalaman konsol baru

Amazon GameLift ServersKonsol baru mencakup peningkatan ini:

- Peningkatan navigasi - Panel navigasi baru memfasilitasi navigasi antar Amazon GameLift Servers sumber daya.
- Amazon GameLift Servershalaman arahan — Halaman arahan baru menyediakan tautan ke dokumentasi yang bermanfaat, menampilkan ikhtisar tingkat tinggiAmazon GameLift Servers, dan memberikan dukungan melalui tautan ke dokumentasi, pertanyaan yang sering diajukan, dan AWS re:Post.
- CloudWatch Metrik Amazon yang disempurnakan — Amazon GameLift Servers metrik sekarang tersedia di Amazon GameLift Servers konsol dan dasbor Anda CloudWatch. Pembaruan ini juga mencakup metrik baru untuk kinerja, pemanfaatan, dan sesi pemain.

Pelajari lebih lanjut:

- [Lacak hosting game di Amazon GameLift Servers konsol](#)
- [Membangun FlexMatch mak comblang](#)

14 Februari 2023: Amazon GameLift Servers sekarang mendukung enkripsi sisi server untuk topik Amazon SNS

Server Side Encryption ((SSE)) untuk topik SNS mengenkripsi data sensitif Anda saat istirahat. SSE menggunakan AWS Key Management Service (AWS KMS) kunci untuk melindungi konten topik SNS Anda.

Pelajari lebih lanjut:

- [Atur notifikasi kejadian untuk penempatan sesi game](#)
- [FlexMatchcara perjudohan](#)
- [Enkripsi saat istirahat](#)

9 Februari 2023: SDK Amazon GameLift Servers server mendukung .NET 6 dengan C #10

Versi SDK yang diperbarui: Server SDK 5.0.0 untuk.NET 6. Tidak diperlukan pembaruan SDK.

Jika Anda menggunakan Unity Real-Time Development Platform, terus gunakan Amazon GameLift Servers server SDK 5.0.0 dengan.NET 4.6. Unity tidak mendukung .NET 6.

Pelajari lebih lanjut:

- Unduh versi terbaru SDK Amazon GameLift Servers server saat [Amazon GameLift Serversmemulai](#)
- [C# server SDK 5.x untuk Amazon GameLift Servers -- Tindakan](#)

31 Januari 2023: SDK Amazon GameLift Servers server mendukung bahasa Go

Versi SDK yang diperbarui: Server SDK 5.0.0 untuk Go

Pelajari lebih lanjut:

- Unduh versi terbaru SDK Amazon GameLift Servers server saat [Amazon GameLift Serversmemulai](#)
- [Go server SDK untuk Amazon GameLift Servers -- Tindakan](#)

1 Desember 2022: Amazon GameLift Servers meluncurkan Amazon GameLift Servers Anywhere dan Amazon GameLift Servers Server SDK 5.0

Versi SDK yang diperbarui: AWS SDK 1.10.21, Server SDK 5.0.0 untuk C ++ dan C #

Amazon GameLift ServersDi mana saja menggunakan sumber daya server game Anda untuk meng-host server Amazon GameLift Servers game. Anda dapat menggunakan Amazon GameLift Servers Anywhere untuk mengintegrasikan sumber daya komputasi Anda sendiri dengan EC2 komputasi Amazon GameLift Servers terkelola untuk mendistribusikan server game Anda di beberapa jenis

komputasi. Anda juga dapat menggunakan Amazon GameLift Servers Anywhere untuk menguji server game secara iteratif tanpa mengunggah build Amazon GameLift Servers untuk setiap iterasi.

Sorotan:

- Armada New Amazon GameLift Servers Anywhere dan tipe komputasi
- Amazon GameLift Servers Di mana saja menghitung pendaftaran sumber daya
- Peningkatan siklus iterasi pengujian

Amazon GameLift Servers Server SDK 5.0.0 memperkenalkan perbaikan pada SDK server yang ada dan jenis sumber daya baru, komputasi. Server SDK 5.0.0 mendukung Amazon GameLift Servers Anywhere dan penggunaan sumber daya komputasi Anda sendiri untuk hosting server game.

Pelajari lebih lanjut:

- [Server SDK 5.x untuk Amazon GameLift Servers](#)
- [Lokasi armada](#)
- [Pilih sumber daya komputasi untuk armada terkelola](#)
- [Buat sebuah Amazon GameLift Servers Armada di mana saja](#)

25 Agustus 2022: Amazon GameLift Servers meluncurkan dukungan untuk Local Zones

Versi SDK yang diperbarui: AWS SDK 1.9.333

Amazon GameLift Servers Sekarang tersedia di delapan Local Zones di Amerika Serikat, sehingga Anda dapat menyebarkan armada Anda lebih dekat ke pemain. Anda dapat menggunakan semua Amazon GameLift Servers fitur terkelola dengan Local Zones dengan menambahkan Local Zones ke armada Anda.

Local Zones memperluas AWS sumber daya dan layanan ke tepi cloud, dekat dengan populasi besar, industri, dan pusat teknologi informasi (TI). Ini berarti Anda dapat menerapkan aplikasi yang memerlukan latensi milidetik satu digit lebih dekat ke pengguna akhir atau ke pusat data lokal.

Pelajari lebih lanjut:

- [Amazon GameLift Servers zona lokal](#)
- [Lokasi armada](#)

- [Buat EC2 armada Amazon GameLift Servers terkelola](#)

28 Juni 2022: Amazon GameLift Servers meluncurkan pengalaman konsol keikutsertaan baru

Amazon GameLift ServersKonsol baru mencakup peningkatan ini:

- Peningkatan navigasi - Panel navigasi baru memfasilitasi navigasi antar Amazon GameLift Servers sumber daya.
- Amazon GameLift Servershalaman arahan — Halaman arahan baru menyediakan tautan ke dokumentasi yang bermanfaat, menampilkan ikhtisar tingkat tinggiAmazon GameLift Servers, dan memberikan dukungan melalui tautan ke dokumentasi, pertanyaan yang sering diajukan, dan AWS re:Post.
- CloudWatch Metrik Amazon yang disempurnakan — Amazon GameLift Servers metrik sekarang tersedia di Amazon GameLift Servers konsol dan dasbor Anda CloudWatch. Pembaruan ini juga mencakup metrik baru untuk kinerja, pemanfaatan, dan sesi pemain.

Pelajari lebih lanjut:

- [Lacak hosting game di Amazon GameLift Servers konsol](#)
- [Membangun FlexMatch mak comblang](#)

15 Februari 2022: FlexMatch menambahkan aturan majemuk dan perbaikan tambahan

FlexMatchpengguna sekarang memiliki akses ke fitur-fitur berikut:

- Aturan majemuk — Menambahkan dukungan untuk aturan perbandingan gabungan untuk pertandingan dari 40 pemain atau lebih sedikit. Anda sekarang dapat menggunakan pernyataan logis untuk membuat aturan majemuk untuk membentuk kecocokan. Tanpa aturan gabungan dalam set aturan Anda, untuk membentuk kecocokan, semua aturan dalam set aturan harus benar. Dengan aturan gabungan, Anda dapat memilih aturan mana yang akan diterapkan menggunakan operator logis berikut:and,or,not, danxor.
- Pemilihan tim yang fleksibel — Ekspresi properti perbandingan yang diperbarui untuk mendukung pemilihan subset dari semua tim yang tersedia.
- Daftar string yang lebih panjang - Meningkatkan jumlah string maksimum dari 10 menjadi 100 dalam daftar string nilai atribut pemain.

Pelajari lebih lanjut:

- [Amazon GameLift ServersFlexMatchpanduan pengembang](#):
 - [FlexMatchjenis aturan](#)
 - [FlexMatchekspresi properti](#)
- [AttributeValue: SL](#)

28 Oktober 2021: Amazon GameLift Servers menambahkan dukungan untuk armada Multi-region di Wilayah Asia Pasifik (Osaka); Amazon GameLift ServersFleetIQ menambahkan dukungan untuk AWS prosesor Graviton2

[Versi SDK yang diperbarui: AWS SDK 1.9.133](#)

Amazon GameLift Serverssekarang tersedia di Wilayah Asia Pasifik (Osaka). Pengembang game sekarang dapat menyebarkan instans di Osaka menggunakan armada GameLift Multi-wilayah.

Anda sekarang dapat menggunakan server game yang dihosting Graviton2, berdasarkan arsitektur prosesor berbasis ARM, untuk mencapai peningkatan kinerja dengan biaya lebih rendah jika dibandingkan dengan opsi komputasi berbasis Intel yang setara.

Sorotan:

- Amazon GameLift Serverssekarang tersedia di Wilayah Asia Pasifik (Osaka).
- Amazon GameLift ServersFleetIQgrup server game sekarang dapat dikonfigurasi untuk mengelola keluarga instance Graviton2 c6g, m6g, dan r6g.

Pelajari lebih lanjut:

- [Amazon GameLift ServersArmada multi-wilayah](#)
- [CreateGameServerGroup](#)
- [AWS prosesor graviton](#)

20 September 2021: Amazon GameLift Servers merilis plugin untuk Unity

Amazon GameLift ServersPlugin untuk Unity versi 1.0.0 berisi pustaka dan UI asli yang membuatnya lebih mudah untuk mengakses Amazon GameLift Servers sumber daya dan mengintegrasikan

Amazon GameLift Servers ke dalam game Unity Anda. Anda dapat menggunakan Amazon GameLift Servers plugin Unity untuk mengakses Amazon GameLift Servers APIs dan menyebarkan AWS CloudFormation template untuk skenario game umum. Plugin ini juga menyertakan contoh permainan yang bekerja dengan skenario sampel. Anda dapat menggunakan Amazon GameLift Servers Local untuk melihat pesan yang diteruskan antara klien game dan server game untuk mempelajari bagaimana permainan biasa berinteraksi dengan Amazon GameLift Servers.

Plugin untuk Unity mendukung Unity 2019.4 LTS dan 2020.3 LTS.

Sorotan:

- Membangun, menjalankan, dan memodifikasi contoh permainan dengan skenario yang berbeda, atau membuat sendiri.
- Terapkan AWS CloudFormation skenario sampel untuk skenario permainan biasa termasuk auth saja, Armada wilayah tunggal, armada multi-wilayah dengan antrian dan mak comblang khusus, Armada Spot dengan antrian dan mak comblang khusus, dan. FlexMatch

Pelajari lebih lanjut:

- [Mengintegrasikan game dengan Amazon GameLift Servers plugin untuk Unity](#)

30 Juni 2021: FlexMatch menambahkan aturan BatchDistance

Anda dapat menggunakan jenis aturan batchDistance untuk menentukan string atau atribut numerik, membawa sejumlah manfaat ke setiap segmen.

Sorotan:

- Untuk pertandingan besar (>40 pemain), alih-alih menyeimbangkan pemain secara merata hanya dengan keterampilan, Anda sekarang bisa mendapatkan keseimbangan yang sama berdasarkan keterampilan, mode, dan peta. Pastikan bahwa semua orang dalam pertandingan berada dalam band keterampilan, band beberapa atribut numerik seperti liga atau gaya bermain, dan kelompok sesuai dengan atribut string seperti peta atau mode permainan. Anda juga dapat membuat ekspansi dari waktu ke waktu. Misalnya, Anda dapat membuat ekspansi untuk mengizinkan rentang tingkat keterampilan yang lebih besar untuk memasuki match seiring dengan semakin lama pemain menunggu.

Untuk match di bawah 40 pemain, Anda dapat menggunakan ekspresi aturan baru yang disederhanakan.

3 Juni 2021: pembaruan SDK klien Amazon GameLift Servers realtime dan SDK server

Versi SDK yang diperbarui: Realtime Client SDK 1.2.0, Server SDK 3.4.0 untuk Unreal

Dengan pembaruan SDK terbaru ini, Anda sekarang dapat mengintegrasikan IL2 CPP ke dalam aplikasi seluler Anda yang menggunakan SDK Klien RTS dan mengikuti praktik terbaik dengan kerangka kerja. Anda juga sekarang dapat membangun Amazon GameLift Servers Server SDK untuk Unreal Version 4.26. Pembaruan ini berisi komponen yang terintegrasi dengan server game Windows atau Linux Anda, termasuk versi C ++ dan C # dari Amazon GameLift Servers Server SDK, Amazon GameLift Servers Lokal, dan plugin Unreal Engine.

Sorotan:

- Menambahkan dukungan untuk IL2 CPP di SDK Klien RTS dan untuk membangun pustaka asli sebagai kerangka kerja, sehingga Anda dapat membangun klien RTS untuk perangkat seluler terbaru.
- Anda dapat menggunakan [DescribePlayerSessions\(\)](#) untuk mendapatkan informasi untuk satu sesi pemain, untuk semua sesi pemain dalam sesi game, atau untuk semua sesi pemain yang terkait dengan satu ID pemain.
- Anda dapat menggunakan [GetInstanceCertificate\(\)](#) untuk mengambil lokasi file dari sertifikat TLS yang dikodekan PEM yang terkait dengan armada dan instance-nya.
- Membuat SDK Server dukungan untuk Unreal versi 4.26.
- C# SDK yang ada, versi 4.0.2, telah diverifikasi kompatibel dengan Unity 2020.3. Tidak ada pembaruan SDK yang diperlukan.

Pelajari lebih lanjut:

- [Amazon GameLift ServersPanduan Pengembang](#):
 - [DescribePlayerSessions\(\)](#)
 - [GetInstanceCertificate\(\)](#)

23 Maret 2021: Amazon GameLift Servers menambahkan pemberitahuan ke penempatan sesi game

[Versi SDK yang diperbarui: AWS SDK 1.8.168](#)

Sekarang Anda dapat menggunakan kejadian untuk memantau aktivitas penempatan sesi game untuk antrian sesi game. Buat topik Amazon Simple Notification Service (Amazon SNS) untuk mempublikasikan pemberitahuan acara, atau mengatur pelacakan peristiwa menggunakan Acara. CloudWatch

Sorotan:

- Untuk setiap antrian, Anda dapat mengatur string teks kustom untuk disertakan dalam semua pesan kejadian.
- Saat menggunakan topik Amazon SNS, Anda dapat mengatur syarat akses tambahan yang membatasi publikasi hanya untuk antrian tertentu.

Pelajari lebih lanjut:

- Amazon GameLift ServersPanduan Pengembang:
 - [Atur notifikasi kejadian untuk penempatan sesi game](#) (baru)
 - [Acara penempatan sesi permainan](#) (baru)
- [Referensi API \(AWS SDK\)](#)
 - Parameter antrian sesi permainan baru `NotificationTarget` dan `CustomEventData`: [GameSessionQueue](#), [CreateGameSessionQueue](#) [UpdateGameSessionQueue](#)
- [Amazon GameLift Serversforum](#)

16 Maret 2021: Amazon GameLift Servers menambahkan armada multi-wilayah, enam wilayah baru

[Versi SDK yang diperbarui: AWS SDK 1.8.163](#)

Amazon GameLift Servershosting terkelola sekarang tersedia di 21 AWS Wilayah. Wilayah baru meliputi Cape Town (af-south-1), Bahrain (me-south-1), Hong Kong (ap-east-1), Milan (eu-south-1), Paris (eu-west-3), dan Stockholm (eu-north-1).

Dengan fitur armada Amazon GameLift Servers multi-lokasi baru, Anda sekarang dapat mengatur satu armada untuk Amazon GameLift Servers meng-host server game Anda di salah satu atau semua 20 Wilayah yang didukung (kecuali Wilayah Beijing). Fitur ini bertujuan untuk secara signifikan mengurangi pekerjaan yang diperlukan untuk mengatur dan memelihara sumber daya Amazon GameLift Servers hosting secara global. Armada multi-lokasi dapat dibuat di AWS Wilayah berikut:

us-east-1 (Virginia N.), (Oregon), us-west-2 (Frankfurt), (eu-west-1 Irlandia), (Sydney), ap-southeast-2 (Tokyo), dan ap-northeast-1 (Seoul). eu-central-1 ap-northeast-2 Di semua Wilayah lain, Anda dapat terus menyiapkan armada lokasi tunggal sesuai kebutuhan. Semua armada yang dibuat sebelum rilis ini adalah armada satu lokasi. Menggunakan armada multi-lokasi tidak memengaruhi biaya hosting Anda. Amazon GameLift Servers Penetapan harga didasarkan pada jenis, lokasi, dan volume instance yang Anda gunakan. (Untuk informasi lebih lanjut, lihat [Amazon GameLift Servers harga](#).) AWS CloudFormation dukungan untuk armada multi-lokasi akan segera tersedia.

Note

Armada multi-lokasi tidak tersedia di Wilayah Tiongkok. Amazon GameLift Servers sumber daya yang berada di Wilayah Tiongkok tidak dapat berinteraksi atau digunakan oleh sumber daya di Wilayah lain Amazon GameLift Servers.

Sorotan:

- Dengan armada multi-lokasi, secara eksplisit menambahkan daftar lokasi terpencil. Amazon GameLift Servers menyebarkan instance dengan tipe dan konfigurasi yang sama, termasuk konfigurasi build dan runtime, ke Wilayah asal armada dan semua lokasi yang ditambahkan.
- Sesuaikan pengaturan kapasitas dan penskalaan untuk setiap lokasi secara independen. Kebijakan penskalaan otomatis berlaku untuk seluruh armada, namun Anda dapat mengaktifkan atau menonaktifkannya berdasarkan lokasi.
- Mulai sesi game baru di lokasi armada tertentu. Saat menggunakan antrian sesi permainan atau perjodohan untuk menempatkan sesi permainan, Anda sekarang dapat memprioritaskan di mana sesi permainan baru dimulai berdasarkan lokasi, biaya hosting, dan latensi pemain.
- Dapatkan metrik hosting di Amazon GameLift Servers konsol, digabungkan untuk semua lokasi dalam armada atau dipecah oleh setiap lokasi armada.

Pelajari lebih lanjut:

- [Blog teknologi game Amazon](#)
- [Referensi API \(AWS SDK\)](#)
 - Operasi lokasi armada baru: [CreateFleetLocations](#), [DescribeFleetLocationAttributes](#), [DescribeFleetLocationCapacity](#), [DescribeFleetLocationUtilization](#), [DeleteFleetLocations](#)

- Operasi armada yang diperbarui, dengan dukungan multi-lokasi baru: [CreateFleet](#), [UpdateFleetCapacity](#), [Jelaskan EC2 InstanceLimits](#), [DescribeInstances](#), [StopFleetActionsStartFleetActions](#)
- Operasi penempatan sesi game yang diperbarui, dengan prioritas baru dan kemampuan penyaringan: [CreateGameSessionQueue](#), [DescribeGameSessionQueuesUpdateGameSessionQueue](#)
- Operasi pembuatan sesi game yang diperbarui, dengan dukungan lokasi baru: [CreateGameSession](#), [DescribeGameSessions](#), [DescribeGameSessionDetails](#), [SearchGameSessions](#)
- [Amazon GameLift ServersPanduan Pengembang](#):
 - [Amazon GameLift Serverslokasi layanan](#) (diperbarui)
 - [Sesuaikan Amazon GameLift Servers EC2 armada yang dikelola](#) (baru)
 - [Menskalakan kapasitas hosting game dengan Amazon GameLift Servers](#) (diperbarui)
 - [Kustomisasi antrian sesi permainan](#) (baru)
 - [Detail armada di Amazon GameLift Servers konsol](#) (diperbarui)
- [Amazon GameLift Serversforum](#)

9 Februari 2021: Amazon GameLift Servers memperluas dukungan untuk instans AMD, mandiri FlexMatch

[Versi SDK yang diperbarui: AWS SDK 1.8.139](#)

Rilis ini mencakup pembaruan berikut:

- Amazon GameLift ServersFleetIQgrup server game sekarang dapat dikonfigurasi untuk mengelola keluarga instance AMD C5a, M5a, dan R5a. Jenis EC2 instans Amazon yang didukung, seperti yang tercantum untuk GameServerGroup [InstanceDefinition](#), sekarang menyertakan yang berikut:
 - 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

Catatan: Instans AMD untuk FleetIQ saat ini tidak tersedia untuk digunakan di Wilayah Tiongkok (Beijing) AWS . Lihat [Ketersediaan fitur dan perbedaan implementasi](#) di Tiongkok.

- Amazon GameLift Servers hosting game terkelola sekarang mendukung instans AMD di Wilayah Tiongkok (Beijing), yang dioperasikan oleh Sinnet. Keluarga Instans AMD baru mencakup M5a dan R5a. Jenis EC2 instans yang didukung, seperti yang tercantum untuk armada [InstanceType](#), sekarang mencakup yang berikut:
 - m5a.large, m5a.xlarge, m5a.2xlarge, m5a.4xlarge, m5a.8xlarge, m5a.12xlarge, m5a.16xlarge, m5a.24xlarge
 - r5a.large, r5a.xlarge, r5a.2xlarge, r5a.4xlarge, r5a.8xlarge, r5a.12xlarge, r5a.16xlarge, r5a.24xlarge
- Amazon GameLift Servers FlexMatch sekarang dapat digunakan sebagai solusi perijodohan mandiri di Wilayah Tiongkok (Beijing), yang dioperasikan oleh Sinnet. Pelanggan dapat membuat FlexMatch mak comblang di Wilayah Beijing dan mengonfigurasi [FlexMatchMode](#) parameter ke STANDALONE. Untuk informasi lebih lanjut tentang FlexMatch, baik dengan hosting Amazon GameLift Servers terkelola atau dengan solusi Amazon GameLift Servers non-hosting, di [Panduan Amazon GameLift Servers FlexMatch Pengembang](#).
- Saat menyiapkan notifikasi acara Amazon GameLift Servers FlexMatch, Anda sekarang dapat menetapkan topik Amazon SNS FIFO sebagai target notifikasi. Untuk informasi selengkapnya, lihat:
 - [MatchmakingConfiguration NotificationTarget](#), Referensi Amazon GameLift Servers API
 - [Mengatur pemberitahuan FlexMatch acara](#), Panduan Amazon GameLift Servers FlexMatch Pengembang
 - [Memperkenalkan Amazon SNS FIFO - First-in-first-out Pub/Sub pesan](#), Blog Berita AWS

22 Desember 2020: SDK Amazon GameLift Servers server mendukung Unreal Engine 4.25 dan Unity 2020

Versi SDK yang diperbarui: Amazon GameLift Servers Server SDK 4.0.2, Unreal plugin versi 3.3.3

Versi terbaru Amazon GameLift Servers Server SDK berisi komponen-komponen berikut:

- Plugin Unreal yang diperbarui telah diperbarui untuk kompatibilitas dengan Unreal Engine 4.25. API tidak diubah.
- C# SDK yang ada, versi 4.0.2, telah diverifikasi kompatibel dengan Unity 2020. Tidak ada pembaruan SDK yang diperlukan.

Unduh versi terbaru Amazon GameLift Servers Server SDK saat [Amazon GameLift Servers](#) memulai.

24 November 2020: Amazon GameLift Servers FlexMatch sekarang tersedia untuk game yang diselenggarakan di mana saja

Versi SDK yang diperbarui: AWS [SDK 1.8.95](#)

Amazon GameLift Servers FlexMatch adalah layanan perijodohan yang dapat disesuaikan untuk game multipemain. Awalnya dirancang untuk pengguna hosting Amazon GameLift Servers terkelola, sekarang FlexMatch dapat diintegrasikan ke dalam game yang menggunakan sistem hosting lain, termasuk peer-to-peer, komputasi lokal berpemilik, dan tipe primitif komputasi awan. Game yang digunakan Amazon GameLift Servers FleetIQ untuk hosting game di Amazon sekarang EC2 dapat menerapkan perijodohan dengan FlexMatch

FlexMatch menyediakan algoritme perijodohan yang kuat dan bahasa aturan yang memberi Anda garis lintang lebar untuk menyesuaikan proses perijodohan sehingga pemain dicocokkan berdasarkan karakteristik pemain utama dan latensi yang dilaporkan. Selain itu, FlexMatch menawarkan alur kerja permintaan perijodohan yang mendukung fitur seperti pesta pemain, penerimaan pemain, dan pengisian ulang pertandingan. Saat Anda menggunakan FlexMatch hosting Amazon GameLift Servers terkelola atau Amazon GameLift Servers Realtime, mak comblang secara otomatis menggunakan Amazon GameLift Servers untuk menemukan sumber daya hosting dan memulai sesi permainan baru untuk pertandingan yang baru dibentuk. Saat menggunakan FlexMatch sebagai layanan mandiri, mak comblang memberikan hasil pertandingan kembali ke game Anda, yang kemudian dapat memulai sesi permainan baru menggunakan solusi hosting Anda.

Operasi API untuk FlexMatch merupakan bagian dari API Amazon GameLift Servers layanan, yang disertakan dalam AWS SDK dan AWS Command Line Interface (AWS CLI). Rilis ini mencakup pembaruan ini untuk mendukung matchmaking mandiri:

- Sumber daya API MatchmakingConfiguration memiliki perubahan berikut:
 - Properti baru, FlexMatchMode menunjukkan apakah mak comblang digunakan dengan hosting Amazon GameLift Servers terkelola atau sebagai perijodohan mandiri.
 - Properti GameSessionQueueArns tidak diperlukan saat FlexMatchMode diatur ke mandiri.
 - Properti ini tidak digunakan dengan matchmaking mandiri: AdditionalPlayerCount, BackfillMode, GameProperties, GameSessionData.
- Fitur backfill otomatis tidak tersedia dengan matchmaking mandiri.

24 November 2020: Instans AMD sekarang tersedia di Amazon GameLift Servers

Versi SDK yang diperbarui: AWS [SDK 1.8.95](#)

Daftar jenis EC2 instans Amazon yang didukung Amazon GameLift Servers sekarang mencakup tiga keluarga instans baru: C5a, M5a, dan R5a. Keluarga ini terdiri dari instans yang dioptimalkan komputasi AMD yang didukung oleh prosesor AMD EPYC yang berjalan pada frekuensi hingga 3,3 GHz. Instans AMD kompatibel dengan x86; game yang saat ini berjalan Amazon GameLift Servers dapat digunakan ke jenis instans AMD tanpa perubahan. Contoh baru tersedia di AWS Wilayah berikut: AS Timur (Virginia N. dan Ohio), AS Barat (Oregon dan California N.), Kanada Tengah (Montreal), Amerika Selatan (Sao Paulo), EU Tengah (Frankfurt), Uni Eropa Barat (London dan Irlandia), Asia Pasifik Selatan (Mumbai), Asia Pasifik Timur Laut (Seoul dan Tokyo), dan Asia Pasifik Tenggara (Singapura dan Sydney).

Instans AMD baru meliputi:

- 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

Pelajari lebih lanjut:

- [Blog teknologi game Amazon](#)
- [Amazon GameLift Serversharga contoh](#)
- [EC2 Instans Amazon yang menampilkan prosesor AMD EPYC](#)
- [Amazon GameLift Serversforum](#)

11 November 2020: Pembaruan versi ke SDK Amazon GameLift Servers server

Versi SDK yang diperbarui: Amazon GameLift Servers Server SDK 4.0.2

Server SDK versi 4.0.2 yang baru memperbaiki masalah yang diketahui dengan operasi API. `StartMatchBackfill()` Operasi ini sekarang mengembalikan respon yang benar untuk permintaan pencocokan isi ulang.

Masalah tidak memengaruhi proses backfill match, dan tidak ada perubahan di cara kerja fitur ini. Masalah ini mungkin memengaruhi pesan log dan penanganan kesalahan untuk permintaan backfill match.

Unduh versi terbaru Amazon GameLift Servers Server SDK saat [Amazon GameLift Servers memulai](#).

5 November 2020: Kustomisasi FlexMatch algoritme baru

FlexMatch pengguna sekarang dapat menyesuaikan perilaku default berikut untuk proses perjodohan. Penyesuaian ini diatur dalam rangkaian aturan matchmaking. Tidak ada perubahan pada Amazon GameLift Servers SDKs.

- Prioritaskan tiket backfill: Anda dapat memilih untuk menaikkan atau menurunkan bagaimana tiket backfill match diprioritaskan ketika mencari match yang dapat diterima. Memprioritaskan tiket isi ulang berguna saat fitur pengisian ulang otomatis diaktifkan. Gunakan properti algoritme `backfillPriority`.
- Pra-sortir untuk mengoptimalkan konsistensi dan efisiensi kecocokan: Konfigurasi mak comblang Anda untuk mengurutkan kumpulan tiket sebelum mengelompokkan tiket untuk dievaluasi. Dengan melakukan pra-sortir tiket berdasarkan atribut utama pemain, hasil match Anda cenderung memiliki pemain yang lebih mirip dalam atribut tersebut. Anda juga dapat meningkatkan efisiensi dalam proses evaluasi dengan melakukan pra-sortir pada atribut yang sama yang digunakan dalam aturan match. Gunakan properti algoritme `sortByAttributes` dengan properti `strategy` diatur ke "disortir".
- Sesuaikan cara waktu tunggu ekspansi dipicu: Pilih antara pemicuan ekspansi berdasarkan usia tiket terbaru (default) atau tertua dalam match yang belum selesai. Memicu tiket tertua cenderung menyelesaikan match lebih cepat, sementara memicu tiket terbaru mengarah ke kualitas match yang lebih tinggi. Gunakan properti algoritme `expansionAgeSelection`.

17 September 2020: Amazon GameLift Servers memperbarui SDK server

Versi SDK yang diperbarui: Amazon GameLift Servers Server SDK 4.0.1

SDK Server baru berisi pembaruan berikut:

- API C# versi 4.0.1
 - Operasi API [TerminateGameSession\(\)](#) tidak lagi didukung. Ganti dengan panggilan ke [ProcessEnding\(\)](#) untuk mengakhiri sesi game dan proses server.
 - Masalah yang diketahui dengan operasi [GetInstanceCertificate\(\)](#) telah diperbaiki.

- Operasi [GetTerminationTime\(\)](#) sekarang mengembalikan nilai tipe data `AwsDateTimeOutcome`.
- API C++ versi 3.4.1
 - Operasi [TerminateGameSession\(\)](#) tidak lagi didukung. Ganti dengan panggilan [ProcessEnding\(\)](#) untuk mengakhiri sesi permainan dan proses server.
- Unreal Plugin Engine versi 3.3.2
 - Operasi [TerminateGameSession\(\)](#) tidak lagi didukung. Ganti dengan panggilan [ProcessEnding\(\)](#) untuk mengakhiri sesi permainan dan proses server.
 - Operasi callback `OnUpdateGameSession` ditambahkan untuk mendukung [FProcessParameter](#) pencocokan isi ulang.

Unduh versi terbaru Amazon GameLift Servers Server SDK saat [Amazon GameLift Servers](#) memulai.

27 Agustus 2020: Amazon GameLift Servers FleetIQ untuk hosting game dengan Amazon EC2 (ketersediaan umum)

Versi SDK yang diperbarui: AWS [SDK 1.8.36](#)

Amazon GameLift Servers FleetIQ Solusi untuk hosting game berbasis cloud berbiaya rendah di Amazon sekarang tersedia secara EC2 umum. Amazon GameLift Servers FleetIQ memberi pengembang kemampuan untuk meng-host server game langsung di Amazon EC2 Spot Instances dengan mengoptimalkan kelangsungan hidup mereka untuk hosting game. Pengembang game dapat menggunakan Amazon GameLift Servers FleetIQ dengan game baru atau untuk menambah kapasitas untuk game yang ada. Solusi ini mendukung penggunaan kontainer atau AWS layanan lain seperti AWS Shield dan Amazon Elastic Container Service (Amazon ECS).

Rilis ketersediaan umum ini mencakup pembaruan berikut untuk Amazon GameLift Servers FleetIQ solusi:

- Operasi API baru `DescribeGameServerInstances` mengembalikan informasi, termasuk status, pada semua instance aktif untuk grup server Amazon GameLift Servers FleetIQ game.
- Strategi penyeimbangan baru, `ON_DEMAND_ONLY`, mengonfigurasi grup server game untuk menggunakan Instans Sesuai Permintaan saja. Anda dapat memperbarui strategi penyeimbangan grup server game kapan saja, sehingga memungkinkan untuk beralih antara menggunakan Instans Spot dan Instans Sesuai Permintaan sesuai kebutuhan.
- Elemen pratinjau berikut telah diluncurkan untuk ketersediaan umum:

- Penggunaan tombol sortir kustom untuk sumber daya server game. Server game dapat diurutkan berdasarkan stempel waktu pendaftaran.
- Penandaan untuk sumber daya server game.

16 April 2020: Amazon GameLift Servers memperbarui SDK server untuk Unity dan Unreal Engine

Versi SDK yang diperbarui: Amazon GameLift Servers Server SDK 4.0.0, Local 1.0.5 Amazon GameLift Servers

Versi terbaru Amazon GameLift Servers Server SDK berisi komponen yang diperbarui berikut:

- C# SDK versi 4.0.0 diperbarui untuk Unity 2019.
- Unreal plugin versi 3.3.1 diperbarui untuk Unreal Engine versi 4.22, 4.23, dan 4.24.
- Amazon GameLift Servers Versi lokal 1.0.5 diperbarui untuk menguji integrasi yang menggunakan SDK server C # versi 4.0.0.

Unduh versi terbaru Amazon GameLift Servers Server SDK saat [Amazon GameLift Servers memulai](#).

2 April 2020: Amazon GameLift Servers FleetIQ tersedia untuk hosting game di EC2 (pratinjau publik)

[Versi SDK yang diperbarui: AWS SDK 1.7.310](#)

Amazon GameLift Servers FleetIQ Fitur ini mengoptimalkan kelangsungan hidup Instans Spot berbiaya rendah untuk digunakan dengan hosting game. Fitur ini sekarang diperluas untuk pelanggan yang ingin mengelola sumber daya hosting mereka secara langsung daripada melalui Amazon GameLift Servers layanan yang dikelola. Solusi ini mendukung penggunaan kontainer atau AWS layanan lain seperti AWS Shield dan Amazon Elastic Container Service (Amazon ECS).

Pelajari lebih lanjut:

[GameTech posting blog](#) di Amazon GameLift Servers FleetIQ

19 Desember 2019: Peningkatan manajemen AWS sumber daya untuk Amazon GameLift Servers sumber daya

[Versi SDK yang diperbarui: AWS SDK 1.7.249](#)

Anda sekarang dapat memanfaatkan alat manajemen AWS sumber daya dengan Amazon GameLift Servers sumber daya. Secara khusus, semua Amazon GameLift Servers sumber daya utama — build, skrip, armada, antrian sesi game, konfigurasi perjodohan, dan set aturan perjodohan — sekarang diberi nilai Amazon Resource Name (ARN). Sumber daya ARN menyediakan pengidentifikasi konsisten yang unik di semua Wilayah. AWS Mereka dapat digunakan untuk membuat kebijakan izin khusus sumber daya AWS Identity and Access Management (IAM). Sumber daya sekarang diberi ARN dan juga pengidentifikasi sumber daya yang sudah ada sebelumnya, yang tidak spesifik Wilayah.

Selain itu, Amazon GameLift Servers sumber daya sekarang mendukung penandaan. Anda dapat menggunakan tag untuk mengatur sumber daya, membuat kebijakan izin IAM untuk mengelola akses ke grup sumber daya, menyesuaikan rincian AWS biaya, dll. Saat mengelola tag untuk Amazon GameLift Servers sumber daya, gunakan tindakan Amazon GameLift Servers API `TagResource()`, `UntagResource()`, dan `ListTagsForResource()`.

Pelajari lebih lanjut:

- [TagResource](#) di Referensi API Amazon GameLift Servers
- [Penandaan sumber daya AWS](#) dalam Referensi Umum AWS
- [Nama sumber daya Amazon](#) di Referensi AWS Umum

14 November 2019: AWS CloudFormation Template baru, pembaruan di Wilayah Tiongkok (Beijing)

Versi SDK yang diperbarui: AWS [SDK 1.7.210](#)

AWS CloudFormation template untuk Amazon GameLift Servers

Amazon GameLift Servers sumber daya sekarang dapat dibuat dan dikelola melalui AWS CloudFormation. Templat AWS CloudFormation build dan fleet yang ada telah diperbarui agar selaras dengan sumber daya saat ini, dan template baru sekarang tersedia untuk skrip, antrian, konfigurasi perjodohan, dan kumpulan aturan perjodohan. AWS CloudFormation template sangat menyederhanakan tugas mengelola kelompok AWS sumber daya terkait, terutama saat menyebarkan game di beberapa Wilayah.

Pelajari lebih lanjut:

- [Amazon GameLift Servers referensi tipe sumber daya](#) di Panduan AWS CloudFormation Pengguna

- [Mengelola Amazon GameLift Servers sumber daya hosting menggunakan AWS CloudFormation](#) di Panduan Amazon GameLift Servers Pengembang

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.