aws

Panduan Developer

Amazon GameLift

Amazon GameLift: Panduan Developer

Table of Contents

Apa yang dimaksud dengan Amazon GameLift?

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

Penggunaan Amazon GameLift

Amazon GameLift mendukung kasus penggunaan ini dan lainnya:

- Gunakan server game multipemain khusus Anda sendiri, atau gunakan server ready-to-go Realtime untuk meng-host game Anda.
- Jalankan sumber daya hosting berbiaya rendah menggunakan Instans Spot Amazon Elastic Compute Cloud (Amazon EC2).
- Secara otomatis menskalakan jumlah sumber daya hosting yang dibutuhkan game Anda berdasarkan penggunaan.
- Kelola sumber daya komputasi Amazon EC2 Anda di satu tempat menggunakan Amazon GameLift FleetiQ.
- Cocokkan pemain dalam game multipemain dengan Amazon GameLiftFlexMatch.
- Uji secara iteratif server game dan build klien Anda dengan Amazon. GameLift Anywhere
- Gunakan perangkat keras Anda sendiri sambil mengelola semuanya di satu tempat dengan Amazon GameLiftAnywhere.

> **Tip**
>
> Untuk mencoba hosting server GameLift game Amazon, lihatMemulai dengan Amazon GameLift.

Memulai dengan GameLift solusi Amazon

GameLiftSolusi Amazon untuk pengembang game

- GameLiftHosting Amazon untuk server khusus
- GameLiftHosting Amazon dengan Server Realtime
- Amazon GameLift FleetiQ untuk hosting di Amazon EC2
- Amazon GameLift FlexMatch untuk perjodohan
- Hosting GameLift Anywhere perangkat keras Amazon

GameLiftHosting Amazon untuk server khusus

Amazon GameLift menggantikan pekerjaan yang diperlukan untuk meng-host server game kustom Anda sendiri. Kemampuan penskalaan otomatis membantu Anda menghindari membayar lebih banyak sumber daya daripada yang Anda butuhkan. Auto scaling juga membantu memastikan bahwa Anda selalu memiliki permainan yang tersedia untuk pemain baru untuk bergabung dengan menunggu minimal.

Untuk informasi selengkapnya tentang GameLift hosting Amazon, lihatCara GameLift kerja Amazon.

Fitur kunci

- Gunakan fitur GameLift manajemen Amazon, termasuk penskalaan otomatis, antrean multi-lokasi, dan penempatan sesi game.
- Deploy server game agar berjalan pada sistem operasi Amazon Linux atau Server Windows.
- Mengelola sesi game dan sesi pemain.
- Siapkan pelacakan kesehatan yang disesuaikan untuk proses server guna mendeteksi masalah dan untuk menyelesaikan proses yang berkinerja buruk.
- Kelola sumber daya game Anda menggunakan AWS CloudFormation template untuk AmazonGameLift.

GameLiftHosting Amazon dengan Server Realtime

Gunakan Server Realtime untuk mendukung game yang tidak memerlukan server game custom-built. Solusi server ringan ini menyediakan server game yang dapat Anda konfigurasikan agar sesuai dengan game Anda.

Untuk informasi selengkapnya tentang GameLift hosting Amazon dengan Server Realtime, lihatMengintegrasikan game dengan Amazon GameLift Realtime Server.

Fitur kunci

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

Amazon GameLift FleetiQ untuk hosting di Amazon EC2

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

Untuk informasi tentang cara kerja Amazon GameLift FleetiQ dengan Amazon EC2 dan EC2 Auto Scaling untuk hosting game, lihat Panduan Pengembang Amazon FleetiQ. GameLift

Fitur kunci

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

Amazon GameLift FlexMatch untuk perjodohan

Gunakan FlexMatch untuk membuat set aturan khusus untuk menentukan kecocokan multipemain untuk game Anda. FlexMatchmenggunakan set aturan untuk membandingkan pemain yang

kompatibel untuk setiap pertandingan dan memberikan pemain dengan pengalaman multiplayer yang ideal.

Untuk informasi selengkapnyaFlexMatch, lihat Apa itu Amazon GameLiftFlexMatch?

Fitur kunci

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

Hosting GameLift Anywhere perangkat keras Amazon

Gunakan Amazon GameLift Anywhere untuk mengintegrasikan perangkat keras di mana saja di lingkungan Anda ke dalam hosting GameLift game Amazon Anda. Anda dapat mengintegrasikan Anywhere armada dan armada EC2 dalam antrian matchmaker dan sesi game untuk mengelola perjodohan dan penempatan game di seluruh perangkat keras Anda.

Untuk informasi lebih lanjut tentang pengujian denganAnywhere, lihatUji integrasi Anda menggunakan GameLift Anywhere armada Amazon. Untuk informasi lebih lanjut tentang menyiapkan Anywhere armada, lihatMenyiapkan GameLift armada Amazon.

Fitur kunci

- Lakukan pengujian cepat dan berulang pada server game dan build klien Anda.
- Gunakan GameLift alat Amazon yang ditetapkan untuk menyebarkan game ke perangkat keras Anda sendiri.
- Gunakan perangkat keras yang paling dekat dengan pemain Anda, di mana saja.

Mengakses Amazon GameLift

Gunakan alat ini untuk bekerja dengan AmazonGameLift.

GameLiftSDK Amazon

> Amazon GameLift SDK berisi pustaka yang diperlukan untuk berkomunikasi dengan Amazon GameLift dari klien game, server game, dan layanan game Anda. Untuk informasi selengkapnya, lihat Dukungan pengembangan dengan Amazon GameLift.

SDK Klien Amazon GameLift Realtime

Realtime Client SDK memungkinkan klien game untuk terhubung ke server Realtime, bergabung dengan sesi game, dan tetap sinkron dengan pemain lain. Unduh SDK dan pelajari lebih lanjut cara melakukan panggilan API dengan API klien Server Realtime (C #).

GameLiftKonsol Amazon

Gunakan AWS Management Consolefor Amazon GameLift untuk mengelola penerapan game Anda, mengonfigurasi sumber daya, dan melacak penggunaan pemain dan metrik kinerja. GameLiftKonsol Amazon menyediakan alternatif GUI untuk mengelola sumber daya secara terprogram dengan (). AWS Command Line Interface AWS CLI

AWS CLI

Gunakan alat baris perintah ini untuk melakukan panggilan ke AWS SDK, termasuk Amazon GameLift API. Untuk informasi tentang penggunaanAWS CLI, lihat Memulai dengan AWS CLI di Panduan AWS Command Line Interface Pengguna.

Harga untuk Amazon GameLift

Amazon GameLift mengenakan biaya untuk instans berdasarkan durasi penggunaan, dan untuk bandwidth menurut jumlah data yang ditransfer. Untuk daftar lengkap biaya dan harga AmazonGameLift, lihat GameLiftHarga Amazon.

Untuk informasi tentang menghitung biaya hosting game Anda atau perjodohan dengan AmazonGameLift, lihatMenghasilkan perkiraan GameLift harga Amazon, yang menjelaskan cara menggunakan. AWS Pricing Calculator

Cara GameLift kerja Amazon

Topik ini mencakup komponen inti untuk hosting game dan menjelaskan bagaimana Amazon GameLift membuat server game multipemain Anda tersedia untuk pemain.

Siap mempersiapkan game Anda untuk hosting di AmazonGameLift? MemeriksaPeta jalan hosting yang GameLift dikelola Amazon.

Komponen kunci

Menyiapkan Amazon GameLift untuk meng-host game Anda melibatkan bekerja dengan komponen-komponen berikut. Diagram dalam Arsitektur game dengan Amazon terkelola GameLift memvisualisasikan hubungan antara komponen-komponen ini.

- Server game adalah perangkat lunak server game Anda yang berjalan di armada. Anda mengunggah build atau skrip server game Anda ke Amazon GameLift dan memberi tahu AmazonGameLift. Saat Anda menggunakan Amazon GameLift Anywhere atau Amazon GameLift FleetiQ, Anda mengunggah build server game Anda langsung ke sumber daya komputasi.
- Sesi permainan adalah permainan yang sedang berlangsung dengan pemain. Anda menentukan karakteristik basic dari sesi game, seperti rentang hidup dan jumlah pemain. Pemain kemudian terhubung ke server game untuk bergabung dengan sesi permainan.
- Klien game adalah perangkat lunak game Anda yang berjalan pada perangkat pemain. Klien game terhubung ke server game melalui layanan backend untuk bergabung dengan sesi game, berdasarkan informasi koneksi yang diterimanya dari AmazonGameLift.
- Layanan backend adalah layanan khusus tambahan yang menangani tugas yang terkait dengan AmazonGameLift. Sebagai praktik terbaik, layanan backend Anda harus menangani semua komunikasi klien game dengan AmazonGameLift.

Server hosting game

Dengan AmazonGameLift, Anda dapat meng-host server game Anda dengan tiga cara berbeda: Amazon TerkelolaGameLift, Amazon GameLift FleetiQ, dan Amazon. GameLift Anywhere Untuk informasi selengkapnya tentang Amazon GameLift FleetiQ, lihat Apa itu Amazon GameLift FleetiQ?

Anda dapat merancang armada yang sesuai dengan kebutuhan game Anda. Untuk informasi lebih lanjut tentang merancang armada, lihatPanduan desain GameLift armada Amazon.

Amazon yang Dikelola GameLift

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

Amazon GameLift Anywhere

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

Alias armada

Alias adalah sebutan yang dapat Anda transfer antar armada, menjadikannya cara yang mudah 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 meluncurkan proses server game pada setiap instans, armada dapat menyelenggarakan sesi game. Amazon GameLift memulai sesi game baru saat layanan klien game Anda mengirimkan permintaan penempatan ke layanan backend atau ke Amazon. GameLift

Penempatan sesi game dan algoritma FleetiQ

Antrian menggunakan algoritma FleetiQ untuk memilih server game yang tersedia untuk menjadi tuan rumah sesi permainan baru. Komponen utama untuk penempatan sesi game adalah antrean sesi GameLift game Amazon. 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. Desain antrean sesi game

Koneksi pemain ke game

Sebagai bagian dari proses penempatan sesi game, antrian atau sesi game meminta server game yang dipilih untuk memulai sesi permainan baru. Server game merespons prompt dan melaporkan kembali ke Amazon GameLift saat siap menerima koneksi pemain. Amazon GameLift 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

Ketika armada aktif dan siap untuk menyelenggarakan 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 duduk diam.

Amazon GameLift menyediakan alat penskalaan otomatis yang sangat efektif, atau Anda dapat mengatur kapasitas armada secara manual. Untuk informasi selengkapnya, lihat Penskalaan kapasitas GameLift hosting Amazon.

Penskalaan otomatis

Amazon GameLift menyediakan dua metode penskalaan otomatis:

- Penskalaan otomatis berbasis target
- Skala otomatis dengan kebijakan berbasis aturan

Fitur penskalaan tambahan

- Perlindungan sesi game - GameLift Mencegah Amazon mengakhiri sesi game yang menjadi tuan rumah pemain aktif selama acara penurunan skala.
- 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 kejadian kapasitas dan penskalaan armada.

Pemantauan Amazon GameLift

Saat Anda memiliki armada yang aktif dan berjalan, Amazon GameLift mengumpulkan 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 mengumpulkan hal-hal berikut:

- Armada, lokasi, sesi permainan, dan detail sesi pemain
- Metrik penggunaan
- Kesehatan proses server
- Log sesi game

Untuk informasi selengkapnya tentang pemantauan di AmazonGameLift, lihatMemantau Amazon GameLift.

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 Akun AWS kelola, secara eksplisit mengizinkan Amazon GameLift mengakses sumber daya Anda. AWS

Amazon GameLift menyediakan beberapa opsi untuk mengelola jenis akses ini. Untuk informasi selengkapnya, lihat Berkomunikasi dengan sumber daya AWS lain dari armada.

Bagaimana pemain terhubung ke game

Sesi permainan adalah instance dari permainan Anda yang berjalan di AmazonGameLift. Untuk memainkan game Anda, seorang 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 menyimpan slot untuk pemain dan memberikan informasi koneksi. Pemain kemudian dapat connect ke sesi game dan mengklaim slot yang dicadangkan.

Untuk informasi rinci tentang membuat dan mengelola sesi permainan dan sesi pemain dengan server game kustom, lihatTambahkan Amazon GameLift ke klien game Anda. Untuk informasi tentang menghubungkan pemain ke Server Realtime, lihatMengintegrasikan klien game untuk Server Realtime.

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

Selenggarakan sesi game dengan sumber daya terbaik yang tersedia di beberapa lokasi

> Pilih dari beberapa opsi saat mengonfigurasi cara Amazon GameLift memilih sumber daya untuk menyelenggarakan sesi game 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.

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 meneruskan data sesi game ke server game saat memulai sesi game baru. Amazon GameLift meneruskan data sesi pemain ke server game saat pemain terhubung ke sesi game.

Filter dan urutkan sesi game 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.

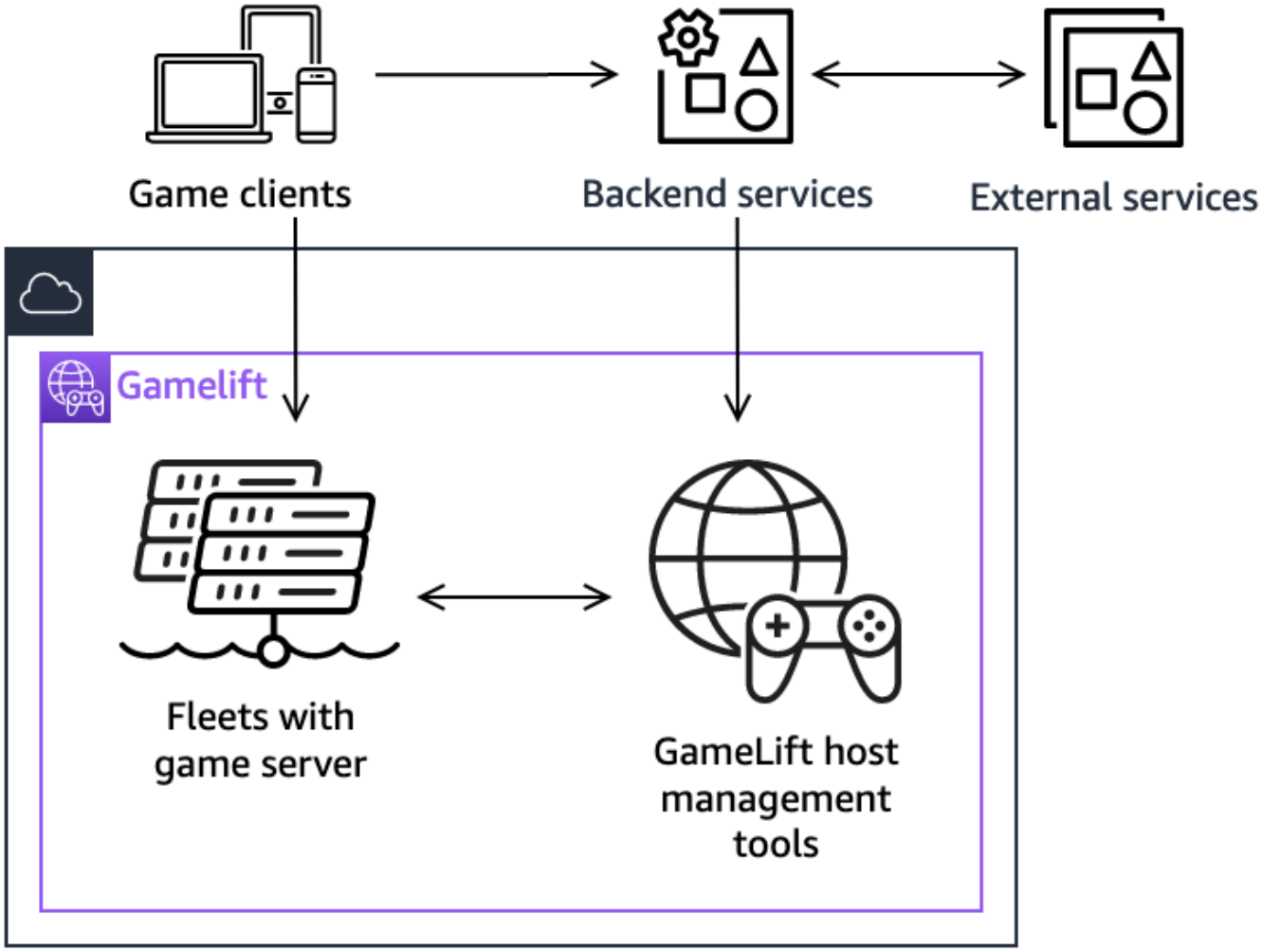
Melacak data penggunaan game dan pemain

Secara otomatis menyimpan log untuk sesi permainan yang telah selesai. Anda dapat mengatur penyimpanan log saat mengintegrasikan Amazon GameLift ke server game Anda. Untuk informasi selengkapnya, lihat Mencatat pesan server di Amazon GameLift.

Gunakan GameLift konsol Amazon untuk melihat informasi terperinci tentang sesi game, termasuk metadata sesi, pengaturan, dan data sesi pemain. Selengkapnya, lihat Melihat data pada sesi game dan pemain dan Metrik.

Arsitektur game dengan Amazon terkelola GameLift

Diagram berikut mengilustrasikan komponen kunci arsitektur game yang dihosting menggunakan GameLift solusi Amazon yang dikelola.



Komponen kunci dari arsitektur ini meliputi:

Klien game

Untuk bergabung dengan game yang dihosting di AmazonGameLift, klien game Anda harus terlebih dahulu menemukan sesi game yang tersedia. Klien game mencari sesi game yang ada, meminta perjodohan, atau memulai sesi game baru dengan berkomunikasi dengan Amazon GameLift melalui layanan backend. Layanan backend membuat permintaan ke AmazonGameLift, dan sebagai tanggapan, layanan menerima informasi sesi game, yang relay kembali ke klien game. Klien game kemudian terhubung ke server game. Untuk informasi selengkapnya, lihat Mempersiapkan game untuk Amazon GameLift.

Layanan backend

Layanan backend menangani komunikasi antara klien game dan Amazon GameLift dengan memanggil operasi API GameLift layanan Amazon di AWS SDK. Anda juga dapat menggunakan layanan backend untuk tugas-tugas khusus game lainnya seperti otentikasi pemain dan otorisasi, inventaris, atau kontrol mata uang. Untuk informasi selengkapnya, lihat Rancang layanan klien game Anda.

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 AmazonGameLift.

Server game

Anda mengunggah perangkat lunak server game Anda ke AmazonGameLift, dan GameLift Amazon menerapkannya ke mesin hosting untuk menyelenggarakan sesi permainan dan menerima koneksi pemain. Server game berkomunikasi dengan Amazon GameLift untuk memulai sesi game, memvalidasi pemain yang baru terhubung, dan melaporkan status sesi game, koneksi pemain, dan sumber daya yang tersedia.

Server game khusus berkomunikasi dengan Amazon GameLift dengan menggunakan Amazon GameLift Server SDK. Klien game terhubung langsung ke server game setelah menerima detail koneksi dari Amazon GameLift melalui layanan backend. Untuk informasi selengkapnya, lihat Integrasikan game dengan server game khusus.

Server realtime adalah server game yang menjalankan skrip khusus Anda. Saat bergabung dengan game, klien game terhubung langsung ke server Realtime menggunakan SDK Klien Realtime. Untuk informasi selengkapnya, lihat Mengintegrasikan game dengan Amazon GameLift Realtime Server.

Alat manajemen host

Saat menyiapkan dan mengelola sumber daya hosting, pemilik game menggunakan alat manajemen hosting untuk mengelola build atau skrip server game, armada, perjodohan, dan antrian. GameLiftAlat Amazon yang disetel di AWS SDK dan konsol menyediakan berbagai cara bagi Anda untuk mengelola sumber daya hosting Anda. Anda dapat mengakses server game individual dari jarak jauh untuk pemecahan masalah.

Mengatur

Dapatkan bantuan dengan menyiapkan Anda Akun AWS untuk menggunakan Amazon untuk GameLift meng-host game multiplayer Anda.

> **Tip**
>
> Untuk mencoba hosting server GameLift game Amazon, lihatMemulai dengan Amazon GameLift.

Topik

Siapkan Akun AWS

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

Topik

Mendaftar Akun AWS

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

Untuk mendaftar Akun AWS

1. Buka https://portal.aws.amazon.com/billing/signup.
2. Ikuti petunjuk secara online.

   Anda akan diminta untuk menerima panggilan telepon dan memasukkan kode verifikasi pada keypad telepon sebagai bagian dari prosedur pendaftaran.

   Saat Anda mendaftar Akun AWS, Pengguna root akun AWS akan dibuat. Pengguna root memiliki akses ke semua Layanan AWS dan sumber daya dalam akun. Sebagai praktik terbaik keamanan, tetapkan akses administratif ke pengguna administratif, dan hanya gunakan pengguna root untuk melakukan tugas yang memerlukan akses pengguna root.

AWS akan mengirimkan email konfirmasi kepada Anda setelah proses pendaftaran selesai. Anda dapat melihat aktivitas akun saat ini dan mengelola akun dengan mengunjungi https://aws.amazon.com/ dan memilih Akun Saya.

Membuat pengguna administratif

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

Mengamankan Pengguna root akun AWS Anda

1. Masuk ke AWS Management Console sebagai pemilik akun dengan memilih Pengguna root dan memasukkan alamat email Akun AWS Anda. Di halaman berikutnya, masukkan kata sandi Anda.

   Untuk bantuan masuk menggunakan pengguna root, lihat Masuk sebagai pengguna root dalam Panduan Pengguna AWS Sign-In.
2. Aktifkan autentikasi multi-faktor (MFA) untuk pengguna root Anda.

   Untuk petunjuknya, silakan lihat Mengaktifkan perangkat MFA virtual untuk pengguna root Akun AWS Anda (konsol) dalam Panduan Pengguna IAM.

Membuat pengguna 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 sebuah pengguna administratif.

   Untuk mendapatkan tutorial tentang menggunakan Direktori Pusat Identitas IAM sebagai sumber identitas Anda, silakan lihat Mengonfigurasi akses pengguna dengan Direktori Pusat Identitas IAM default di Panduan Pengguna AWS IAM Identity Center.

Masuk sebagai pengguna administratif

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

  Untuk bantuan masuk menggunakan pengguna Pusat Identitas IAM, lihat Masuk ke portal akses AWS dalam Panduan Pengguna AWS Sign-In.

Mengelola izin pengguna untuk Amazon GameLift

Buat pengguna tambahan atau perluas izin akses ke pengguna yang ada sesuai kebutuhan untuk GameLift sumber daya Amazon 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

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

Untuk memberikan akses, tambahkan izin ke pengguna, grup, atau peran Anda:

- Pengguna dan grup di AWS IAM Identity Center:

  Buat rangkaian izin. Ikuti petunjuk dalam Buat set izin dalam Panduan Pengguna AWS IAM Identity Center.

- Pengguna yang dikelola di IAM melalui penyedia identitas:

Buat peran untuk federasi identitas. Ikuti petunjuk dalam Membuat peran untuk penyedia identitas pihak ketiga (federasi) di Panduan Pengguna IAM.

- Pengguna IAM:
  - Buat peran yang dapat diambil pengguna Anda. Ikuti petunjuk dalam Membuat peran untuk pengguna IAM dalam Panduan Pengguna IAM.
  - (Tidak disarankan) Pasang kebijakan langsung ke pengguna atau tambahkan pengguna ke grup pengguna. Ikuti petunjuk di 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.

Siapkan akses terprogram untuk pengguna

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

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

Pengguna mana yang membutuhkan akses terprogram?	Untuk	Oleh
Identitas tenaga kerja (Pengguna yang dikelola di Pusat Identitas IAM)	Gunakan kredensial sementara untuk menandatangani permintaan terprogram ke AWS CLI, SDK AWS, atau API AWS.	Mengikuti petunjuk untuk antarmuka yang ingin Anda gunakan. • Untuk AWS CLI, lihat Mengonfigurasi AWS CLI untuk menggunakan AWS IAM Identity Center di Panduan Pengguna AWS Command Line Interface. • Untuk SDK AWS, alat, dan API AWS, lihat Autentika

Pengguna mana yang membutuhkan akses terprogram?	Untuk	Oleh
		si Pusat Identitas IAM di Panduan Referensi SDK dan Alat AWS.
IAM	Gunakan kredensial sementara untuk menandatangani permintaan terprogram ke AWS CLI, SDK AWS, atau API AWS.	Mengikuti petunjuk dalam Menggunakan kredensial sementara dengan sumber daya AWS di Panduan Pengguna IAM.
IAM	(Tidak direkomendasikan) Gunakan kredensial jangka panjang untuk menandatangani permintaan terprogram ke AWS CLI, SDK AWS, atau API AWS.	Mengikuti petunjuk untuk antarmuka yang ingin Anda gunakan. • Untuk AWS CLI, lihat Mengautentikasi menggunakan kredensial pengguna IAM di Panduan Pengguna AWS Command Line Interface. • Untuk SDK dan alat AWS, lihat Mengautentikasi menggunakan kredensial jangka panjang di Panduan Referensi SDK dan Alat AWS. • Untuk API AWS, 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 menggunakan SDKAWS. Misalnya, Anda menggunakan layanan backend (bertindak atas nama 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

Untuk Amazon GameLift, 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 ke Amazon. GameLift 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 dengan Amazon. GameLift

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

Gunakan sintaks dalam contoh ini untuk menetapkan izin AWS Identity and Access Management (IAM) bagi pengguna yang memerlukan akses ke sumber daya Amazon. GameLift Untuk informasi selengkapnya tentang mengelola izin pengguna, lihatMengelola izin pengguna untuk Amazon GameLift. Saat mengelola izin untuk pengguna di luar IAM Identity Center, sebagai praktik terbaik selalu melampirkan izin ke peran IAM atau grup pengguna, bukan pengguna individual.

Jika Anda menggunakan Amazon GameLift FleetiQ sebagai solusi mandiri, lihat Mengatur Amazon FleetiQ Anda Akun AWS. GameLift

Contoh izin administrator

Contoh-contoh ini memberi pengguna akses penuh untuk mengelola sumber daya hosting GameLift game Amazon.

Example Sintaks untuk izin GameLift sumber daya Amazon

Contoh berikut memperluas akses ke semua GameLift sumber daya Amazon.

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

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

Contoh berikut memperluas akses ke semua GameLift sumber daya Amazon 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 bagian Referensi Umum AWS.

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

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

Contoh berikut memperluas akses ke semua GameLift sumber daya Amazon dan memungkinkan pengguna untuk meneruskan peran layanan IAM ke Amazon. GameLift Peran layanan memberi

Amazon kemampuan GameLift terbatas untuk mengakses sumber daya dan layanan lain atas nama Anda, seperti yang dijelaskan dalamMenyiapkan peran layanan IAM untuk Amazon GameLift. Misalnya, saat menanggapi `CreateBuild` permintaan, Amazon GameLift memerlukan akses ke file build Anda di bucket Amazon S3. Untuk informasi selengkapnya tentang `PassRole` tindakan tersebut, 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 Amazon GameLift API. Mereka mencakup skenario umum untuk mengelola sesi permainan, sesi pemain, dan perjodohan. Untuk rincian lebih lanjut, lihat Siapkan akses terprogram untuk game Anda.

Example Sintaks untuk izin penempatan sesi game

Contoh berikut memperluas akses ke GameLift API Amazon yang menggunakan antrean penempatan sesi game untuk membuat sesi game 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 GameLift API Amazon yang mengelola aktivitas FlexMatch perjodohan. FlexMatchmencocokkan pemain untuk sesi game baru atau yang sudah ada dan memulai penempatan sesi game untuk game yang dihosting di AmazonGameLift. Untuk informasi selengkapnyaFlexMatch, lihat Apa itu Amazon GameLiftFlexMatch?

```
{
  "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 game manual

Contoh berikut memperluas akses ke GameLift API Amazon yang secara manual membuat sesi game 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

Beberapa GameLift fitur Amazon mengharuskan Anda memperluas akses terbatas ke AWS sumber daya 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 serupa dengan pengguna IAM, yang merupakan identitas AWS dengan kebijakan izin yang menentukan apa yang dapat dan tidak dapat dilakukan oleh identitas di 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 GameLift terkelola Amazon Anda. Jika Anda menggunakan Amazon GameLift FleetiQ untuk mengoptimalkan hosting game di instans Amazon Elastic Compute Cloud (Amazon EC2), lihat Mengatur Amazon FleetiQ Anda. Akun AWS GameLift

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

Buat peran IAM kustom

Langkah 1: Buat kebijakan izin.

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 YAML yang dapat digunakan dalam templat AWS CloudFormation.

   Untuk melakukannya, di editor Kebijakan pilih Tindakan, lalu pilih Buat CloudFormation templat. Untuk mempelajari selengkapnya tentang AWS CloudFormation, lihat Referensi jenis sumber daya AWS Identity and Access Management di Panduan Pengguna AWS CloudFormation.
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 tanda di IAM, lihat Menandai sumber daya IAM di Panduan Pengguna IAM.
11. Pilih Buat kebijakan untuk menyimpan kebijakan baru Anda.

Langkah 2: Buat peran yang GameLift dapat diasumsikan Amazon.

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.

Sintaks kebijakan izin

- Izin bagi Amazon GameLift 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"
    }
  ]
}
```

Dukungan pengembangan dengan Amazon GameLift

Amazon GameLift menyediakan serangkaian SDK yang dapat Anda gunakan dengan solusi hosting game terkelola Anda. Gunakan Amazon GameLift SDK untuk menambahkan fungsionalitas yang diperlukan ke server game multipemain, klien game, dan layanan game yang perlu berinteraksi dengan layanan GameLift hosting Amazon.

Untuk informasi terbaru tentang versi Amazon GameLift SDK dan kompatibilitas SDK, lihat. Catatan GameLift rilis Amazon

Untuk server game kustom

Buat dan terapkan server game kustom 64-bit dengan SDK GameLift server Amazon. Server game yang terintegrasi dengan SDK server dan digunakan untuk hosting dapat berkomunikasi dengan GameLift layanan Amazon untuk memulai dan mengelola sesi permainan. Untuk informasi tentang mengintegrasikan SDK server, lihat topik di. Mempersiapkan game untuk Amazon GameLift

Pengembangan sistem operasi

- Windows
- Linux

Bahasa pemrograman yang didukung

Amazon GameLift menyediakan SDK server untuk bahasa berikut. Unduh SDK Server. Untuk informasi spesifik versi terperinci, lihat file Readme yang disertakan di setiap paket.

- Server C++ SDK
  - Referensi SDK
  - Integrasi SDK
- C# server SDK (versi mungkin mendukung .NET 4 dan .NET 6)
  - Referensi SDK
  - Integrasi SDK
- Go
  - Referensi SDK
  - Integrasi SDK

Mesin game yang didukung

Gunakan SDK khusus bahasa dengan mesin apa pun yang mendukung pustaka C++, C#, atau Go. Selain itu, Amazon GameLift menyediakan plugin mesin game ini: Unduh plugin Amazon GameLift

- Kesatuan
  - Plugin SDK server C # untuk Unity adalah plugin ringan dengan pustaka bawaan yang dapat Anda instal menggunakan manajer paket Unity. Gunakan plugin ini dengan versi Unity berikut:

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.

  - Integrasikan Amazon GameLift ke dalam proyek Unity
- Plugin mandiri untuk Unity 2021.3 LTS dan 2022.3 LTS adalah plugin berfitur lengkap dengan pustaka C# SDK yang dibuat untuk elemen Unity dan GUI untuk mengonfigurasi dan menyebarkan sumber daya Amazon untuk hosting. GameLift
  - GameLift Plugin Amazon untuk panduan Unity untuk server SDK 5.x
  - Referensi SDK GameLift server Amazon untuk C #
- Mesin Tidak Nyata
  - Plugin SDK server C ++ untuk Unreal adalah plugin ringan yang terdiri dari kode sumber C++ Unreal yang dapat Anda bangun ke dalam pustaka untuk digunakan dengan Unreal Engine versi 4, 5, dan 5.1.
    - Integrasikan Amazon GameLift ke dalam proyek Unreal Engine
    - Referensi SDK GameLift 5.x server Amazon Unreal Engine
  - Plugin mandiri untuk Unreal Engine 5.0, 5.1, dan 5.2 adalah plugin berfitur lengkap dengan C ++ untuk pustaka SDK server Unreal dan SDK. AWS Plugin dipasang di editor Unreal, dengan elemen UI dan materi pendukung untuk mengonfigurasi dan menyebarkan sumber daya GameLift Amazon untuk hosting.
    - Mengintegrasikan game dengan GameLift plugin Amazon untuk Unreal Engine
    - Referensi SDK GameLift 5.x server Amazon Unreal Engine

Sistem operasi server game

Gunakan Amazon GameLift Server SDK untuk membangun server game agar berjalan di platform berikut:

- Windows Server 2016
- Amazon Linux 2023
- Amazon Linux 2 (AL2)
- Windows Server 2012 (lihat GameLift FAQ Amazon untuk Windows 2012)
- Amazon Linux (AL1) (lihat GameLift FAQ Amazon untuk AL1)

Untuk layanan klien khusus

Buat layanan klien kustom 64-bit menggunakan AWS SDK dengan Amazon GameLift API. SDK ini memungkinkan layanan klien untuk mengelola sesi permainan dan bergabung dengan pemain ke game yang di-host di Amazon GameLift. Untuk memulai, unduh AWS SDK. Untuk informasi selengkapnya tentang penggunaan SDK dengan Amazon GameLift, lihat Referensi GameLift API Amazon.

Untuk Server Realtime

Konfigurasikan dan terapkan server Realtime untuk meng-host game multipemain Anda. Agar klien game dapat terhubung ke server Realtime, gunakan Amazon GameLift Realtime Client SDK. Klien game menggunakan SDK ini untuk bertukar pesan dengan server Realtime dan dengan klien game lain yang terhubung ke server. Untuk memulai, unduh Amazon GameLift Realtime Client SDK. Untuk informasi konfigurasi, lihatMengintegrasikan klien game untuk Server Realtime.

Dukungan SDK

SDK Klien Realtime 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 server Realtime ke sumber daya hosting yang berjalan di platform berikut:

- Amazon Linux
- Amazon Linux 2

Kelola biaya hosting game Anda

AWSTagihan Anda mencerminkan biaya hosting game Anda. Anda dapat melihat perkiraan biaya untuk bulan berjalan, dan biaya akhir untuk bulan-bulan sebelumnya di konsol Penagihan di https://console.aws.amazon.com/billing/. 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.

Secara khusus, pertimbangkan tips ini untuk membantu Anda mengelola biaya GameLift layanan Amazon.

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 untuk Amazon GameLift 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 GameLift armada Amazon

Gunakan tag alokasi AWS biaya untuk mengatur dan melacak biaya hosting game Anda berdasarkan armada Amazon EC2 GameLift Amazon dan sumber daya EC2 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 tampilanAWS Cost Explorer.

Untuk informasi lebih lanjut, lihat topik-topik ini:

- Menggunakan tag alokasi AWS biaya, AWS BillingPanduan Pengguna
- Menganalisis biaya Anda dengan AWS Cost Explorer, Panduan AWS Cost Management Pengguna

Atur kapasitas armada yang tidak terpakai ke nol

Armada dapat terus mengeluarkan biaya bahkan ketika mereka tidak menggunakan sesi permainan hosting. 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.

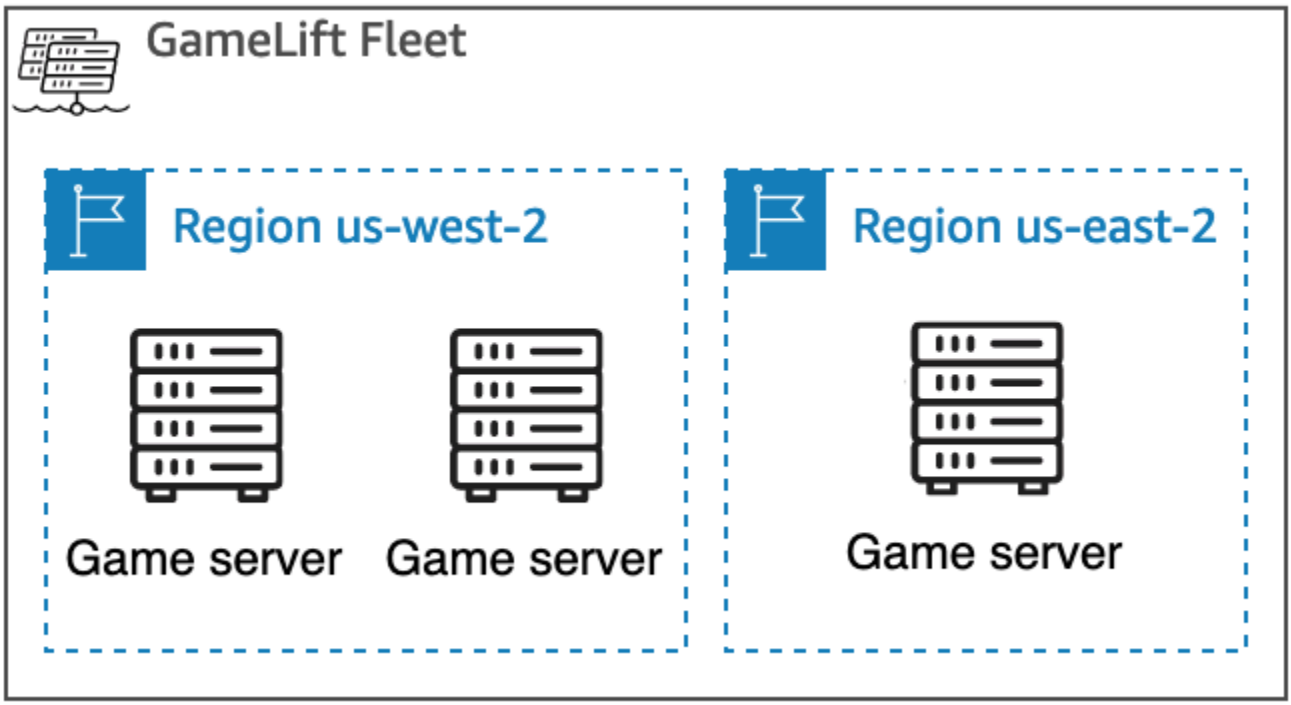
Lokasi GameLift hosting Amazon

Amazon GameLift tersedia di beberapa Wilayah AWS dan Local Zones. Untuk daftar lengkap lokasi, lihat GameLifttitik akhir Amazon dan kuota di. Referensi Umum AWS

GameLift Hosting Amazon

Saat Anda membuat GameLift armada Amazon, Amazon GameLift membuat sumber daya armada saat iniWilayah AWS. Amazon GameLift menyebut Wilayah ini sebagai Wilayah asal armada. Untuk mengelola armada, akses dari daerah asalnya.

Armada multi-lokasi menyebarkan instance ke lokasi lain selain Wilayah asal armada. Dengan armada multi-lokasi, Anda dapat mengelola kapasitas untuk setiap lokasi secara individual, dan Anda dapat menempatkan sesi permainan berdasarkan lokasi. Armada multi-lokasi dapat memiliki lokasi terpencil di Wilayah atau Zona Lokal mana pun yang didukung Amazon GameLift . Diagram berikut menggambarkan armada multi-lokasi dengan sumber daya di dua Wilayah. Dalam diagram, `us-west-2` Wilayah mencakup dua server game, dan `us-east-2` Wilayah memiliki satu server game.



Jika Anda memilih untuk menggunakan armada multi-lokasi dengan instance di Wilayah yang tidak diaktifkan secara default, Anda harus mengaktifkan Wilayah tersebut di Wilayah Anda. Akun AWS Selain itu, kebijakan GameLift administrator Amazon Anda harus mengizinkan `ec2:DescribeRegions` tindakan tersebut. Untuk informasi selengkapnya tentang Wilayah yang tidak diaktifkan secara default dan cara mengaktifkannya, lihat Mengelola Wilayah AWS di Referensi Umum AWS. Untuk contoh kebijakan dengan Wilayah yang tidak diaktifkan secara default, lihatContoh izin administrator.

Important

Untuk menggunakan Wilayah yang tidak diaktifkan secara default, aktifkan di wilayah AndaAkun 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.

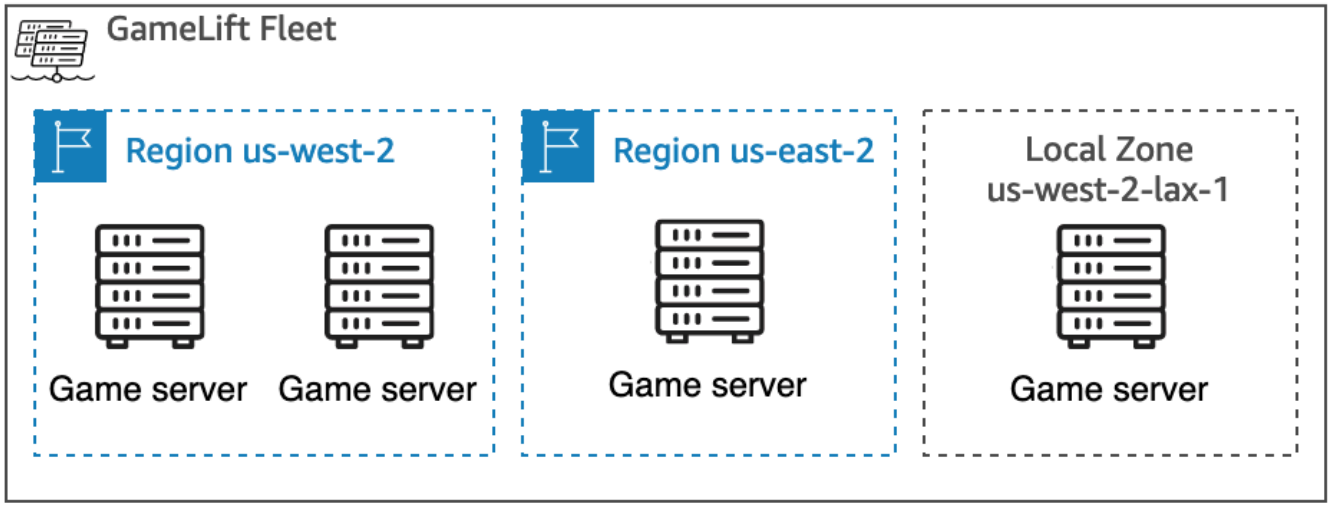
Untuk penempatan sesi game, Anda dapat membuat antrian sesi game di lokasi mana pun yang didukung Amazon GameLift . Amazon GameLift menempatkan sesi permainan dari lokasi tempat Anda membuat antrian.

Zona Lokal

Zona Lokal adalah perpanjangan Wilayah AWS dari kedekatan geografis dengan pengguna Anda. Local Zones memiliki koneksi sendiri ke internet. Local Zones juga mendukung AWS Direct Connect sehingga sumber daya yang dibuat di Local Zone dapat melayani pengguna lokal dengan komunikasi latensi rendah. Lihat informasi selengkapnya di Zona Lokal AWS.

Kode untuk Zona Lokal adalah kode Region, diikuti oleh pengidentifikasi yang menunjukkan lokasi fisiknya. Misalnya, Zona `us-west-2-lax-1` Lokal berada di Los Angeles. Untuk daftar Local Zones yang tersedia, lihatLocal Zones yang Tersedia.

Amazon GameLift menyelenggarakan game Anda di setiap lokasi yang Anda pilih untuk armada Anda. Diagram berikut menggambarkan armada dengan dua server game di `us-west-2` Wilayah, satu server game di `us-east-2` Wilayah, dan satu server game di Zona `us-west-2-lax-1` Lokal.



Local Zones yang Tersedia

Tabel berikut mencantumkan Local Zones yang tersedia dan lokasi fisiknya.

Zona Lokal	Lokasi (area metro)	
`us-east-1-atl-1`	Atlanta	

Zona Lokal	Lokasi (area metro)	
us-east-1-chi-1	Chicago	
us-east-1-dfw-1	Dallas	
us-east-1-iah-1	Houston	
us-east-1-mci-1	Kota Kansas	
us-west-2-den-1	Denver	
us-west-2-lax-1	Los Angeles	
us-west-2-phx-1	Phoenix	

Amazon GameLift Anywhere

Anda dapat menggunakan Amazon GameLift Anywhere untuk membuat armada dengan perangkat keras Anda sendiri, dan mengelola build game, skrip, server game, dan klien menggunakan Amazon. GameLift Amazon GameLift Anywhere tersedia di semua Wilayah yang GameLift didukung Amazon. Untuk informasi selengkapnya tentang membuat Anywhere armada dan menguji integrasi server game Anda, lihat Membuat GameLift Anywhere armada Amazon danUji integrasi Anda menggunakan GameLift Anywhere armada Amazon.

Dengan Amazon, GameLift Anywhere Anda membuat lokasi khusus yang mewakili lokasi fisik perangkat keras yang Anda gunakan untuk meng-host server game GameLift terintegrasi Amazon Anda.

Amazon GameLift FlexMatch

Untuk FlexMatch, Anda dapat meng-host sesi game yang dihasilkan pertandingan di lokasi mana pun yang didukung Amazon GameLift. Aktivitas perjodohan yang sebenarnya terjadi di Wilayah AWS tempat Anda memilih untuk membuat sumber daya mak comblang Anda. GameLift Rute Amazon mencocokkan permintaan dengan mak comblang dan memprosesnya di lokasi tersebut. Untuk informasi selengkapnya tentang Amazon GameLift FlexMatch, lihat Apa itu Amazon GameLift FlexMatch?

Wilayah AWSyang mendukung FlexMatch sumber daya

Amazon GameLift di China

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

- Amazon Web Services di Indonesia
- Amazon GameLift (Memulai Amazon Web Services di China)

Memulai dengan Amazon GameLift

Kami menyarankan Anda mencoba contoh-contoh berikut sebelum Anda menggunakan Amazon GameLift untuk permainan Anda sendiri. Contoh server game khusus memberi Anda pengalaman dengan hosting game di GameLift konsol Amazon. Contoh Realtime Server menunjukkan cara menyiapkan game untuk hosting menggunakan Server Realtime.

Untuk memulai dengan Amazon GameLift untuk permainan Anda sendiri, lihatPeta jalan hosting yang GameLift dikelola Amazon.

Contoh server game khusus

Contoh ini menunjukkan permainan kustom langsung di AmazonGameLift. Contohnya menuntun Anda melalui langkah-langkah berikut:

- Membuat contoh game build.
- Membuat armada untuk menjalankan server game.
- Menghubungkan ke server game dari klien contoh game.
- Meninjau metrik armada dan sesi game.

Setelah langkah-langkah ini, Anda dapat memulai beberapa klien game dan memainkan game untuk menghasilkan data hosting. Kemudian, Anda dapat menjelajahi GameLift konsol Amazon untuk melihat sumber daya hosting Anda, melacak metrik, dan bereksperimen dengan cara untuk meningkatkan kapasitas hosting.

Untuk memulai, masuk ke GameLiftkonsol Amazon.

Game contoh Server Realtime

Contoh ini adalah game multipemain lengkap bernama Mega Frog Race, dengan kode sumber disertakan. Contoh ini menunjukkan cara mengintegrasikan klien game Anda dengan Server Realtime. Anda juga dapat menggunakan game contoh ini sebagai titik awal untuk bereksperimen dengan GameLift fitur Amazon lainnya sepertiFlexMatch.

Untuk tutorial langsung, lihat Membuat Server untuk Game Seluler Multiplayer dengan Hanya Beberapa Baris JavaScript di Blog AWS Game.

Untuk kode sumber Mega Frog Race, lihat GitHubrepositori.

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 API langsung ke Amazon. GameLift
- Skrip waktu nyata - File skrip sumber yang mengonfigurasi armada Server Realtime untuk game tersebut. Skrip ini mencakup konfigurasi minimum yang diperlukan untuk Server Realtime untuk berkomunikasi dengan Amazon GameLift dan untuk meng-host game.

Peta jalan hosting yang GameLift dikelola Amazon

Topik ini membantu Anda memilih dari berbagai opsi GameLift hosting Amazon untuk game multipemain berbasis sesi Anda. Topik lainnya di bagian ini memandu Anda melalui cara menggunakan Amazon GameLift untuk hosting terkelola Anda.

Sebelum Anda mulai bersiap untuk meluncurkan game Anda ke produksi, isi kuesioner peluncuran untuk mulai bekerja dengan tim AmazonGameLift.

Topik

Pilih opsi hosting

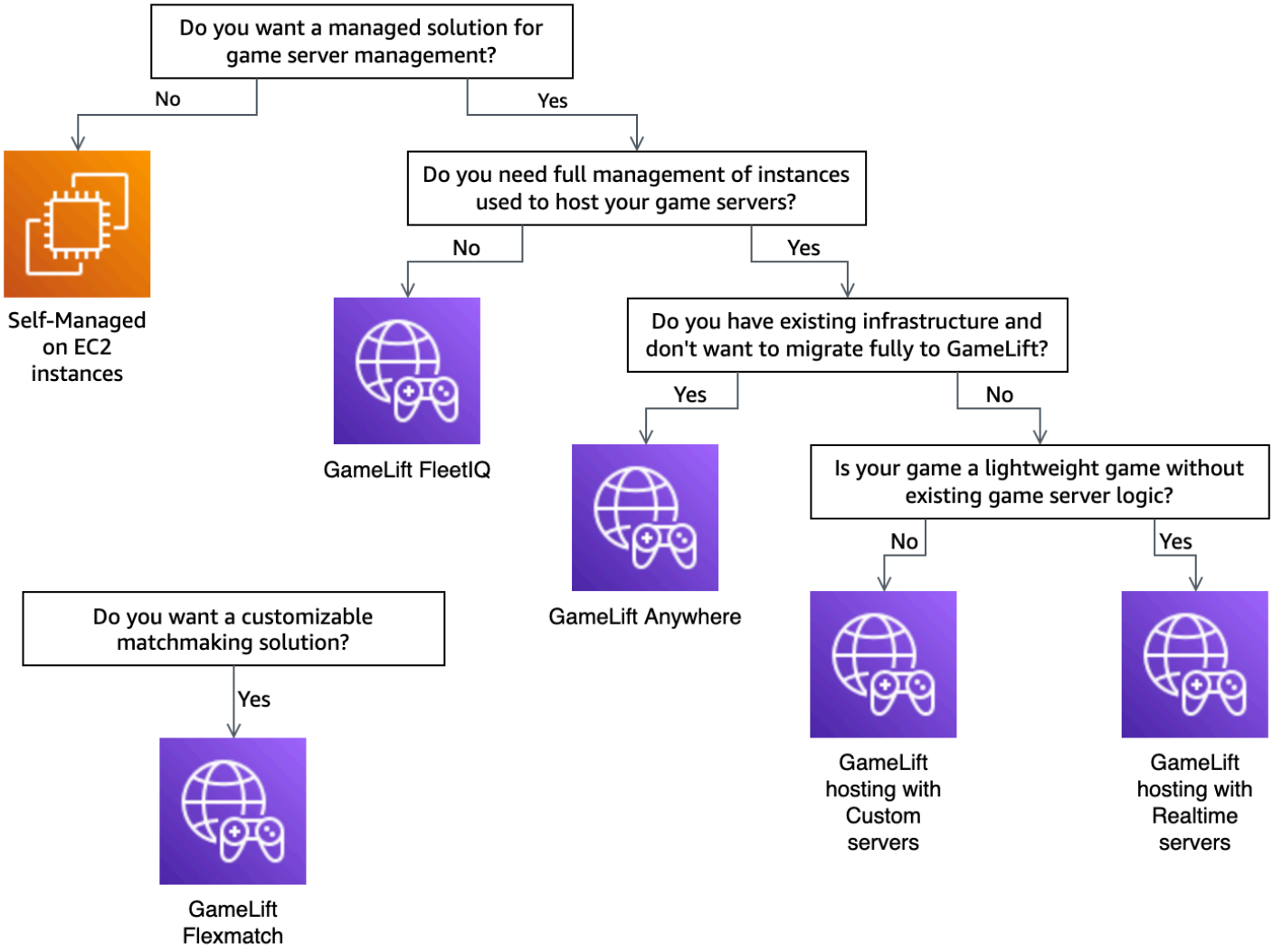
Diagram alur berikut mengajukan pertanyaan untuk mengarahkan Anda ke GameLift solusi Amazon yang benar untuk kasus penggunaan Anda.

1. Apakah Anda menginginkan solusi terkelola untuk manajemen server game?
   - Ya - Lanjutkan ke langkah kedua.
   - Tidak — Pertimbangkan server game yang dikelola sendiri di instans Amazon EC2.
2. Apakah Anda memerlukan kontrol penuh atas instance hosting server game Anda?
   - Ya - Pertimbangkan Amazon GameLift FleetiQ.
   - Tidak - Lanjutkan ke langkah 3.
3. Apakah Anda memiliki infrastruktur yang ingin Anda gunakan dengan AmazonGameLift?
   - Ya - Pertimbangkan Amazon GameLiftAnywhere.

- Tidak - Lanjutkan ke langkah keempat.

4. Apakah game Anda ringan tanpa logika server game yang ada?

   - Ya - Pertimbangkan server Realtime.
   - Tidak - Pertimbangkan server khusus.



Berikut adalah beberapa informasi lebih lanjut tentang beberapa opsi GameLift hosting Amazon yang disebutkan dalam flowchart:

Amazon GameLift Anywhere

Gunakan Amazon GameLift Anywhere untuk meng-host game Anda di perangkat keras Anda sendiri dengan manfaat alat GameLift manajemen Amazon. Anda juga dapat menggunakan

Anywhere armada untuk menguji server game Anda secara berulang. Untuk informasi selengkapnya, lihat Membuat GameLift Anywhere armada Amazon.

Amazon yang Dikelola GameLift

Ada dua opsi untuk GameLift hosting Amazon terkelola:

Server khusus - Amazon GameLift menghosting server khusus Anda yang menjalankan biner server game Anda.

Server Realtime - Amazon GameLift menghosting server game ringan Anda.

GameLiftAmazon

Dalam diagram alur, migrasi lift dan shift mengacu pada migrasi saat Anda tidak dapat membuat perubahan pada arsitektur game. Menggunakan Amazon GameLift FleetiQ memerlukan lebih sedikit perubahan pada penerapan yang ada dan menyediakan GameLift alat Amazon untuk pengelolaan armada. Untuk informasi selengkapnya, lihat Panduan Pengembang Amazon GameLift FleetiQ.

Jika Anda memutuskan untuk menggunakan Amazon GameLift Anywhere atau Amazon yang dikelolaGameLift, lanjutkanSiapkan game Anda untuk Amazon GameLift.

Siapkan game Anda untuk Amazon GameLift

Topik ini menjelaskan langkah-langkah untuk mempersiapkan game multipemain Anda untuk integrasi dengan GameLift hosting Amazon yang dikelola. Untuk mempersiapkan permainan Anda, Anda harus mengaktifkan komunikasi antara itu dan AmazonGameLift.

Siapkan server game khusus Anda

Untuk memulai dan menghentikan sesi permainan, dan untuk melakukan tugas lain, server game harus dapat memberi tahu Amazon GameLift tentang statusnya. Untuk mengaktifkan komunikasi dengan AmazonGameLift, tambahkan kode ke proyek server game Anda. Untuk informasi selengkapnya, lihat Integrasikan game dengan server game khusus.

1. Siapkan server game khusus Anda untuk hosting di AmazonGameLift.

   - Dapatkan Amazon GameLift Server SDK dan buat untuk bahasa pemrograman dan mesin game pilihan Anda.

- Tambahkan kode ke proyek server game Anda untuk mengaktifkan komunikasi dengan AmazonGameLift.

2. Siapkan klien game Anda untuk terhubung ke sesi game yang GameLift dihosting Amazon.

   - Tambahkan AWS SDK ke layanan backend dan proyek klien game Anda. Untuk informasi selengkapnya, lihat Mengunduh Amazon GameLift SDK untuk layanan klien.
   - Tambahkan fungsionalitas untuk mengambil informasi tentang sesi permainan, menempatkan sesi permainan baru, dan cadangan ruang untuk pemain pada sesi permainan.
   - (Opsional) Gunakan FlexMatch untuk perjodohan pemain. Untuk informasi selengkapnya, lihat FlexMatchintegrasi dengan GameLift hosting Amazon.

Siapkan server Realtime

Amazon GameLift Realtime Servers menyediakan solusi server ringan yang dapat Anda konfigurasikan agar sesuai dengan game Anda. Server Realtime memberikan manfaat yang sama dengan yang GameLift ditawarkan Amazon ke server game, tetapi dengan penyesuaian server game yang berkurang.

Buat skrip Realtime untuk hosting di AmazonGameLift.

Skrip realtime berisi konfigurasi server Anda dan logika permainan kustom opsional. Server waktu nyata dibuat untuk memulai dan menghentikan sesi permainan, menerima koneksi pemain, dan mengelola komunikasi dengan Amazon GameLift dan antar pemain dalam permainan. Ada juga kait bagi Anda untuk menambahkan logika server khusus untuk game Anda. Server realtime menggunakan Node.js danJavaScript. Untuk informasi selengkapnya, lihat Membuat skrip Realtime dan Uji integrasi Anda dengan Amazon GameLift.

Uji integrasi Anda dengan Amazon GameLift

Amazon GameLift mendukung iterasi cepat saat menguji server game Anda. Topik ini memandu Anda melalui jenis pengujian yang tersedia.

Server game khusus

Gunakan Amazon GameLift untuk mengintegrasikan perangkat keras di mana saja di lingkungan Anda ke dalam arsitektur hosting GameLift game Amazon Anda. Amazon GameLift Anywhere mendaftarkan perangkat keras Anda dengan Amazon GameLift dalam Anywhere armada, sehingga Anda dapat menguji menggunakan komputer pengembangan lokal Anda sendiri. Untuk informasi

selengkapnya tentang pengujian dengan Amazon GameLiftAnywhere, lihatUji integrasi Anda menggunakan GameLift Anywhere armada Amazon. Untuk informasi selengkapnya tentang menggunakan Amazon GameLift Anywhere untuk menghosting game Anda dengan solusi lokal, lihatMemilih AmazonGameLiftmenghitung sumber daya.

Server Realtime

Dengan Server Realtime, Anda dapat memperbarui skrip kapan saja. Saat Anda memperbarui skrip Realtime, Amazon GameLift mendistribusikan versi baru ke sumber daya hosting Anda dalam beberapa menit. Setelah Amazon GameLift menerapkan skrip baru, semua sesi game baru menggunakan versi skrip baru. Setelah Amazon GameLift menerapkan skrip baru, Anda dapat segera memulai pengujian. Untuk informasi selengkapnya tentang Server Realtime lihat, Mengintegrasikan game dengan Amazon GameLift Realtime Server

Rencanakan dan terapkan sumber daya Amazon GameLift

Gunakan tips berikut untuk membantu merencanakan penyebaran GameLift sumber daya Amazon global Anda. Untuk informasi tentang tempat Anda dapat meng-host game Anda dengan AmazonGameLift, lihatLokasi GameLift hosting Amazon.

Untuk menerapkan GameLift sumber daya Amazon Anda, selesaikan tugas-tugas berikut:

- Paket dan unggah server game Anda ke Amazon GameLift atau ke perangkat keras Anda. Saat mengunggah server Anda ke AmazonGameLift, Anda mengunggahnya hanya ke rumah Wilayah AWS armada Anda. Amazon GameLift secara otomatis mendistribusikan server ke lokasi lain di armada. Untuk informasi selengkapnya, lihat Mengunggah build dan skrip ke Amazon GameLift.
- Rancang dan terapkan GameLift armada Amazon untuk game Anda. Tentukan jenis sumber daya komputasi yang akan digunakan, lokasi mana yang akan digunakan, apakah akan menggunakan antrian, dan opsi lainnya. Untuk informasi selengkapnya, lihat Panduan desain GameLift armada Amazon.
- Buat antrian untuk mengelola penempatan sesi game baru dan strategi Spot Instance. Untuk informasi selengkapnya, lihat Desain antrean sesi game.
- Gunakan penskalaan otomatis untuk mengelola kapasitas hosting armada Anda untuk permintaan pemain yang diharapkan. Untuk informasi selengkapnya, lihat Penskalaan kapasitas GameLift hosting Amazon.
- Gunakan aturan FlexMatch perjodohan untuk permainan Anda. Untuk informasi selengkapnya, lihat FlexMatchintegrasi dengan GameLift hosting Amazon.

Menerapkan sumber daya Amazon GameLift Anda secara otomatis

Untuk menyederhanakan penerapan global GameLift sumber daya Amazon Anda, sebaiknya gunakan infrastruktur sebagai kode (IAC) untuk menentukan sumber daya. Karena Amazon GameLift mendukung AWS CloudFormation template, Anda dapat menetapkan parameter dalam template untuk konfigurasi khusus penerapan apa pun.

Untuk mengelola penyebaran AWS CloudFormation tumpukan Anda, kami juga merekomendasikan penggunaan alat dan layanan integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD) seperti. AWS CodePipeline Ini membantu Anda menyebarkan secara otomatis atau dengan persetujuan setiap kali Anda membangun biner server game.

Berikut ini adalah beberapa langkah umum penyebaran GameLift sumber daya Amazon untuk versi server game baru yang dapat Anda otomatiskan menggunakan alat atau layanan CI/CD:

- Membangun dan menguji biner server game Anda.
- Mengunggah biner ke Amazon GameLift atau perangkat keras Anda.
- Menyebarkan armada baru di gedung baru.
- Setelah Anda menggunakan armada baru, menghapus armada versi sebelumnya dari GameLift antrian Amazon Anda dan menggantinya dengan armada baru.
- Setelah armada versi sebelumnya berhasil mengakhiri semua sesi permainan, menghapus AWS CloudFormation tumpukan armada tersebut.

Anda juga dapat menggunakan AWS Cloud Development Kit (AWS CDK) untuk menentukan GameLift sumber daya Amazon Anda. Untuk informasi selengkapnya tentang AWS CDK, lihat Panduan Developer AWS Cloud Development Kit (AWS CDK).

Rancang layanan klien game Anda

Kami menyarankan Anda menerapkan layanan klien game yang mengautentikasi pemain Anda dan berkomunikasi dengan Amazon GameLift API. Dengan menerapkan layanan klien game kustom, Anda dapat:

- Sesuaikan otentikasi untuk pemain Anda.
- Kontrol cara Amazon GameLift mencocokkan dan memulai sesi game.
- Gunakan database pemain Anda untuk atribut pemain seperti rating keterampilan untuk perjodohan alih-alih mempercayai klien.

Menggunakan layanan klien game juga mengurangi risiko keamanan yang diperkenalkan oleh klien game yang berinteraksi langsung dengan Amazon GameLift API Anda.

Mengautentikasi pemain Anda

Anda dapat menggunakan Amazon Cognito dan ID sesi pemain 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 di dalamnyaAWS. Anda juga dapat menggunakan otorisasi Lambda untuk logika otorisasi kustom dengan API Gateway.

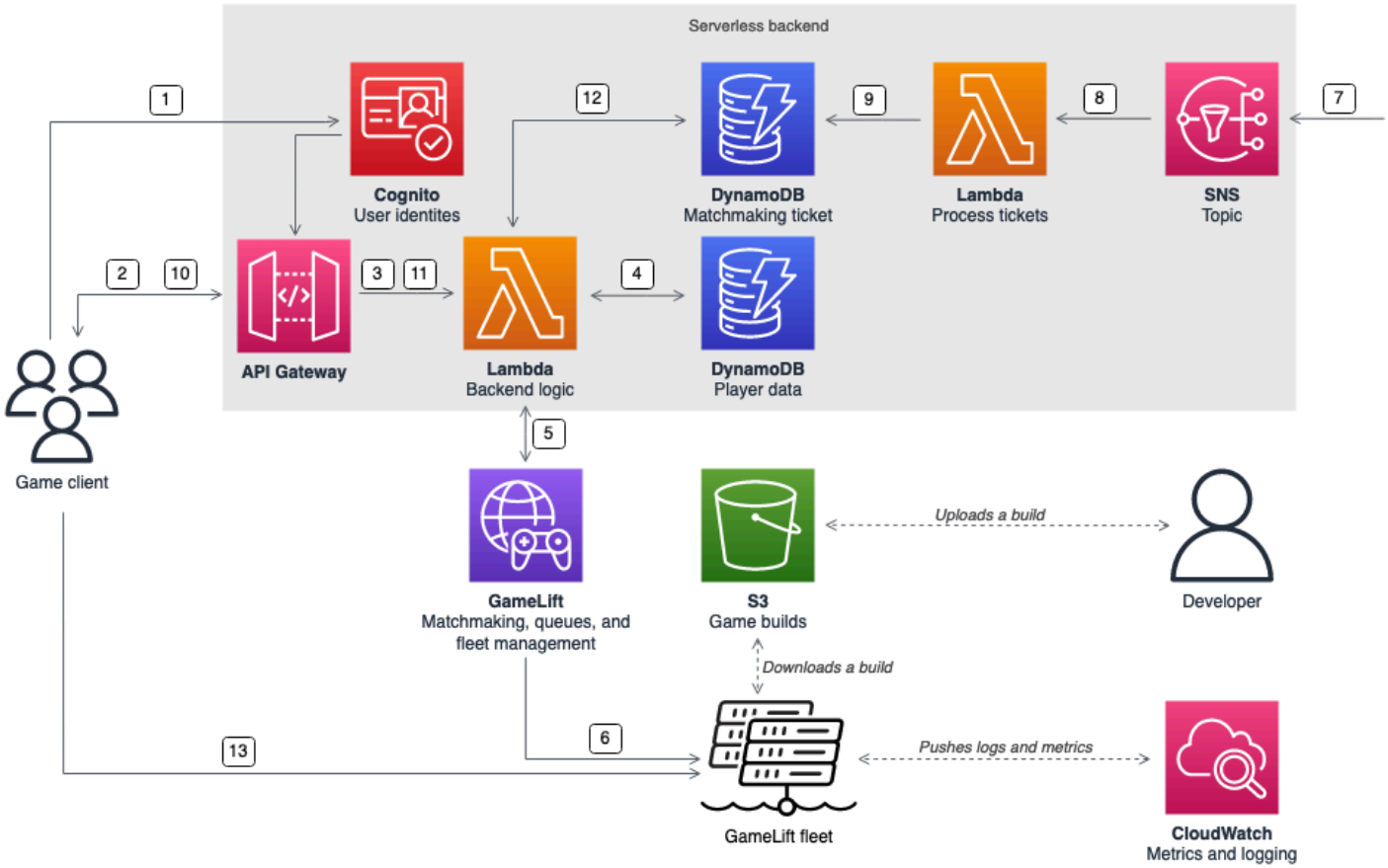
Sumber daya tambahan:

- Menggunakan kumpulan identitas (identitas federasi) (Panduan Pengembang Amazon Cognito)
- Memulai dengan kumpulan pengguna (Panduan Pengembang Amazon Cognito)
- Cara Mengatur Otentikasi Pemain dengan Amazon Cognito (AWSuntuk Blog Game)

Server sesi game mandiri dengan backend tanpa server

Dengan menggunakan arsitektur layanan klien tanpa server, backend dapat melihat status tiket perjodohan dari database yang sangat skalabel alih-alih dengan mengakses API Amazon secara langsung. GameLift

Diagram berikut menunjukkan backend tanpa server yang dibuat dengan Layanan AWS yang cocok pemain ke dalam game yang berjalan di armada Amazon. GameLift Daftar berikut memberikan deskripsi untuk setiap callout bernomor dalam diagram. Untuk mencoba contoh ini, lihat Hosting Game berbasis Sesi Multiplayer aktif. AWS GitHub



1. Klien game meminta identitas pengguna Amazon Cognito dari kumpulan identitas Amazon Cognito.
2. Klien game menerima kredensi akses sementara dan meminta sesi game melalui API Gateway API Amazon.
3. API Gateway memanggil AWS Lambda fungsi.
4. Fungsi Lambda meminta data pemain dari tabel NoSQL Amazon DynamoDB. Fungsi ini menyediakan identitas Amazon Cognito dalam data konteks permintaan.
5. Fungsi Lambda meminta kecocokan melalui perjodohan Amazon GameLiftFlexMatch.
6. FlexMatchmencocokkan sekelompok pemain dengan latensi yang sesuai, dan kemudian meminta penempatan sesi game melalui GameLift antrian Amazon. Antrian memiliki armada dengan satu atau lebih Wilayah AWS lokasi di dalamnya.
7. Setelah Amazon GameLift menempatkan sesi di salah satu lokasi armada, Amazon GameLift mengirimkan pemberitahuan acara ke topik Amazon Simple Notification Service (Amazon SNS).
8. Fungsi Lambda menerima peristiwa Amazon SNS dan memprosesnya.

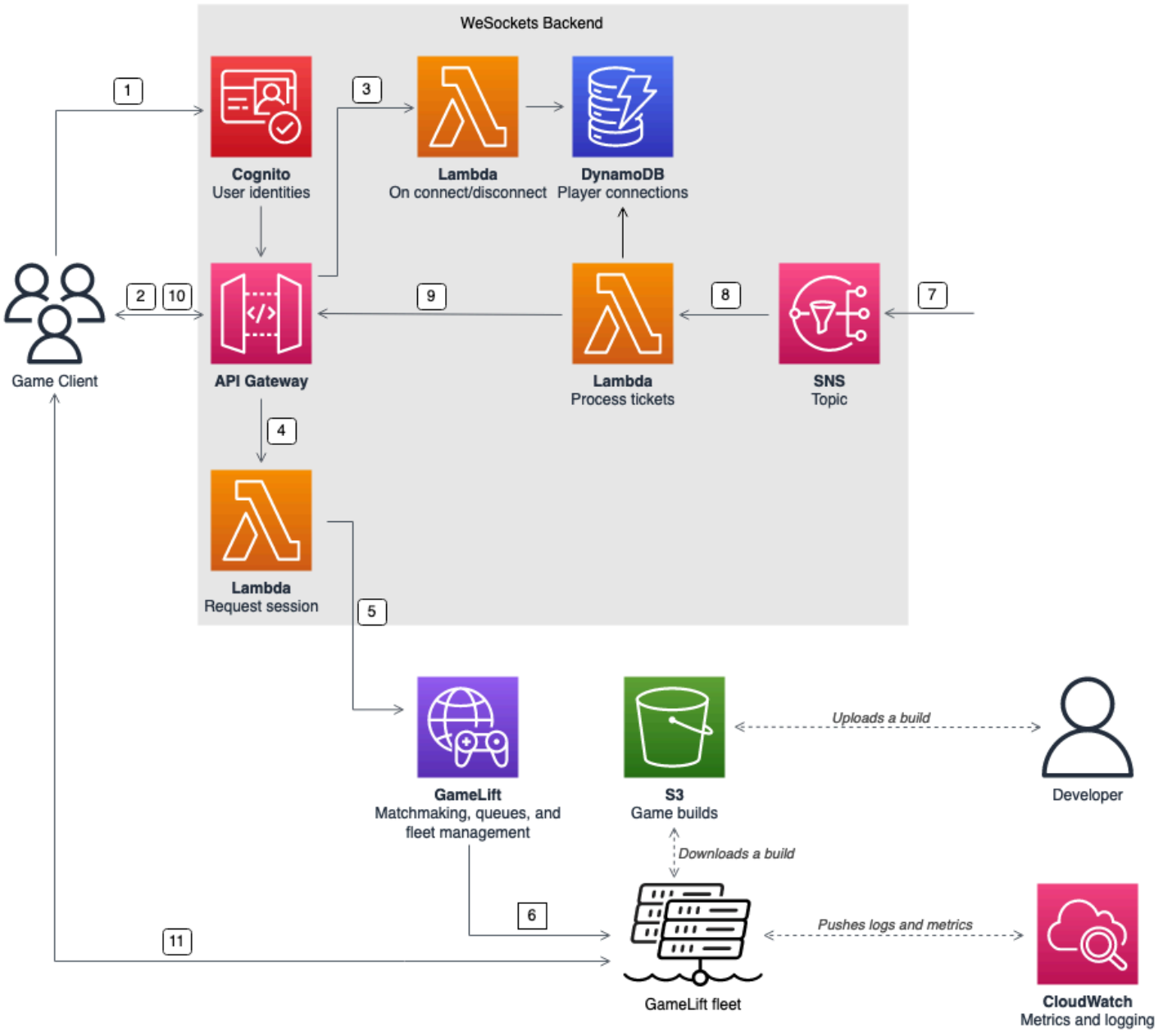
9. Jika tiket perjodohan adalah `MatchmakingSucceeded` peristiwa, maka fungsi Lambda menulis hasilnya, bersama dengan port dan alamat IP server game, ke tabel DynamoDB.
10. Klien game membuat permintaan yang ditandatangani ke API Gateway untuk melihat status tiket perjodohan pada interval tertentu.
11. API Gateway menggunakan fungsi Lambda yang memeriksa status tiket perjodohan.
12. Fungsi 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 akan mengirimkan respons yang memverifikasi bahwa kecocokan belum siap.
13. Klien game terhubung ke server game menggunakan TCP atau UDP dengan menggunakan port dan alamat IP yang disediakan oleh layanan backend. Klien game kemudian mengirimkan ID sesi pemain ke server game, yang kemudian memvalidasi ID menggunakan Amazon GameLift Server SDK.

Server sesi game mandiri dengan backend WebSocket berbasis

Dengan menggunakan arsitektur WebSocket berbasis Amazon API Gateway, Anda dapat membuat permintaan perjodohan dengan WebSockets dan mengirim notifikasi 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 API, lihat Bekerja dengan WebSocket API.

Diagram berikut menunjukkan arsitektur backend WebSocket berbasis yang menggunakan API Gateway dan lainnya Layanan AWS untuk mencocokkan pemain ke dalam game yang berjalan di GameLift armada Amazon. 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 kredensi Amazon Cognito.
3. API Gateway memanggil AWS Lambda fungsi pada koneksi. Fungsi menyimpan informasi koneksi dalam tabel Amazon DynamoDB.
4. Klien game mengirimkan pesan ke fungsi Lambda, melalui API Gateway API melalui WebSocket koneksi, untuk meminta sesi.

5. Fungsi Lambda menerima pesan dan kemudian meminta kecocokan melalui perjodohan Amazon GameLiftFlexMatch.
6. Setelah FlexMatch mencocokkan sekelompok pemain, FlexMatch meminta penempatan sesi permainan melalui GameLift antrian Amazon.
7. Setelah Amazon GameLift menempatkan sesi di salah satu lokasi armada, Amazon GameLift mengirimkan pemberitahuan acara ke topik Amazon Simple Notification Service (Amazon SNS).
8. Fungsi Lambda menerima peristiwa Amazon SNS dan memprosesnya.
9. Jika tiket perjodohan adalah sebuah `MatchmakingSucceeded` event, maka fungsi Lambda meminta koneksi pemain yang benar dari DynamoDB. Fungsi kemudian mengirimkan pesan ke klien game melalui API Gateway API melalui WebSocket koneksi. Dalam arsitektur ini, klien game tidak secara aktif melakukan polling status perjodohan.
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 oleh layanan backend. Klien game juga mengirimkan ID sesi pemain ke server game, yang kemudian memvalidasi ID menggunakan Amazon GameLift Server SDK.

Menyiapkan metrik dan pencatatan log untuk Amazon GameLift

Anda dapat menggunakan data yang dikumpulkan dari server dan sumber daya GameLift game Amazon untuk membantu mengidentifikasi anomali. Anda juga dapat menggunakan metrik untuk membantu meningkatkan kinerja.

Area utama yang harus diamati untuk Amazon GameLift meliputi:

- Metrik GameLift layanan Amazon — Amazon GameLift menyediakan CloudWatch metrik Amazon pada sumber daya Anda termasuk server game, armada, antrean, dan. FlexMatch Anda dapat menemukan metrik ini di GameLift konsol Amazon dan CloudWatch konsol. Untuk informasi selengkapnya tentang GameLift metrik AmazonCloudWatch, lihatPantau Amazon GameLift dengan Amazon CloudWatch.
- Metrik server game — Amazon tidak GameLift dapat mengakses metrik server game Anda. Namun, Anda dapat mengirim metrik khusus CloudWatch langsung dari server game Anda dengan menggunakan CloudWatch agen. Anda juga dapat menggunakan peran armada AWS Identity and Access Management (IAM) dan AWS SDK untuk mengirim metrik secara langsung. CloudWatch Untuk contoh cara mengonfigurasi metrik, lihat Hosting Game berbasis Sesi Multiplayer aktif. AWS

GitHub Repositori ini mencakup contoh konfigurasi CloudWatch agen dan kode untuk klien C # statSD.

- Log server game — Untuk mengonfigurasi file log server game Anda di server game, gunakan konfigurasi Amazon GameLift Server SDK. Anda juga dapat menggunakan Amazon CloudWatch Logs sebagai solusi pengelolaan log waktu nyata, dan Anda dapat mengonfigurasi log dengan CloudWatch agen. Untuk informasi selengkapnya, lihat Mencatat pesan server di Amazon GameLift.

Daftar periksa peluncuran game

Anda dapat menggunakan daftar periksa ini untuk memvalidasi fase penerapan game Anda. Dalam daftar periksa, item yang ditandai [Kritis] sangat penting untuk peluncuran produksi Anda.

Topik

Orientasi

Gunakan daftar periksa berikut untuk melacak item untuk orientasi permainan Anda untuk hosting AmazonGameLift. Item yang ditandai [Critical] sangat penting untuk peluncuran produksi Anda.

- [Kritis] Isi kuesioner GameLift orientasi Amazon di konsol Amazon. GameLift
- [Kritis] Rancang dan terapkan layanan backend untuk klien game untuk berinteraksi dengan server game Anda.
- [Kritis] Buat peran AWS Identity and Access Management (IAM) yang Anda berikan ke instans GameLift server Amazon untuk akses ke sumber daya lainAWS.
- [Kritis] Desain dan menerapkan failover untuk lainnya Wilayah AWS untuk FlexMatch dan antrian.
- Rencanakan peluncuran armada ke lokasi target Anda, mengingat antrian permainan dan struktur armada Anda.
- Otomatiskan penyebaran 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).

Pengujian

Gunakan daftar periksa berikut untuk melacak item pengujian saat mengembangkan game Anda dengan GameLift hosting Amazon. Item yang ditandai [Critical] sangat penting untuk peluncuran produksi Anda.

- [Kritis] Lengkapi kuesioner peluncuran, dan kirimkan kuesioner yang telah selesai ke tim peluncuran Amazon. GameLift Anda dapat menemukan kuesioner peluncuran di konsol Amazon GameLift.
- [Kritis] Permintaan peningkatan untuk kuota GameLift layanan Amazon dan Layanan AWS kuota lainnya sehingga lingkungan hidup Anda dapat meningkatkan kebutuhan produksi.
- [Kritis] Verifikasi bahwa port terbuka pada armada langsung cocok dengan kisaran port yang dapat digunakan server Anda.
- [Kritis] Tutup port RDP 3389 dan port SSH 22.
- Mengembangkan rencana untuk DevOps pengelolaan permainan Anda. Jika Anda menggunakan Amazon CloudWatch Logs atau metrik CloudWatch kustom Amazon, tentukan alarm untuk masalah parah atau kritis pada armada server. Simulasikan kegagalan dan uji runbook.
- Verifikasi bahwa jumlah server yang berjalan pada instans pada penggunaan penuh berada dalam kemampuan jenis instans server.
- Tune kebijakan penskalaan Anda untuk menjadi lebih konservatif pada awalnya dan memberikan kapasitas idle lebih dari yang Anda pikir Anda butuhkan. Anda dapat mengoptimalkan biaya nanti. Pertimbangkan penggunaan kebijakan penskalaan berbasis target dengan kapasitas idle 20 persen.
- Gunakan aturan FlexMatch latensi untuk mencocokkan pemain yang secara geografis dekat sama. Wilayah AWS Uji perilaku dari aturan ini dengan adanya beban dengan data latensi sintetis dari client uji beban Anda.
- Beban uji otentikasi pemain dan infrastruktur sesi permainan Anda untuk melihat apakah skala secara efektif untuk memenuhi permintaan.
- Verifikasi bahwa server yang tersisa berjalan selama beberapa hari masih dapat menerima koneksi.
- Tingkatkan level AWS Support paket Anda ke Bisnis atau Perusahaan sehingga AWS dapat merespons Anda selama masalah atau pemadaman.

Luncurkan

Gunakan daftar periksa berikut untuk melacak item peluncuran untuk game Anda yang dihosting di AmazonGameLift. Item yang ditandai [Critical] sangat penting untuk peluncuran produksi Anda.

- [Kritis] Tetapkan kebijakan perlindungan armada ke perlindungan penuh pada semua armada langsung sehingga penskalaan tidak menghentikan sesi permainan yang aktif.
- [Kritis] Atur 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 pengembang Anda untuk berpartisipasi dalam acara peluncuran dan pantau peluncuran game Anda di ruang peluncuran.
- Pantau latensi pemain dan pengalaman pemain.

Pasca-peluncuran

Gunakan daftar periksa berikut untuk melacak item pasca-peluncuran untuk game Anda yang dihosting di Amazon. GameLift

- Tune aturan penskalaan untuk meminimalkan kapasitas idle.
- Ubah FlexMatch aturan atau tambahkan lokasi tambahan berdasarkan persyaratan latensi Anda.
- Optimalkan server yang dapat dieksekusi, karena efisiensi kinerjanya secara langsung mempengaruhi biaya armada. Untuk menjalankan lebih banyak sesi game dengan infrastruktur yang sama, tingkatkan jumlah proses server per instans.
- Gunakan data analitik Anda untuk mendorong pengembangan berkelanjutan, meningkatkan pengalaman pemain dan umur panjang game, dan mengoptimalkan monetisasi.

Mempersiapkan game untuk Amazon GameLift

Untuk mempersiapkan game multipemain Anda untuk hosting di Amazon GameLift, atur komunikasi antara game Anda dan Amazon GameLift. Topik di bagian ini memberikan bantuan terperinci untuk mengintegrasikan game Anda dengan Amazon GameLift, server game khusus, dan Server Realtime, dan untuk menambahkan perjodohan dengan. FlexMatch

Topik

Integrasikan game dengan server game khusus

Amazon GameLift menyediakan set alat lengkap untuk menyiapkan game multipemain dan server game khusus Anda untuk berjalan di AmazonGameLift. Amazon GameLift SDK berisi pustaka yang diperlukan untuk klien game dan server untuk berkomunikasi dengan Amazon. GameLift Untuk informasi selengkapnya tentang SDK dan tempat mendapatkannya, lihatDukungan pengembangan dengan Amazon GameLift.

Topik di bagian ini berisi petunjuk terperinci tentang cara menambahkan GameLift fungsionalitas Amazon ke klien game dan server game Anda sebelum diterapkan di AmazonGameLift. Untuk peta jalan lengkap agar game Anda aktif dan berjalan di AmazonGameLift, lihat. Peta jalan hosting yang GameLift dikelola Amazon

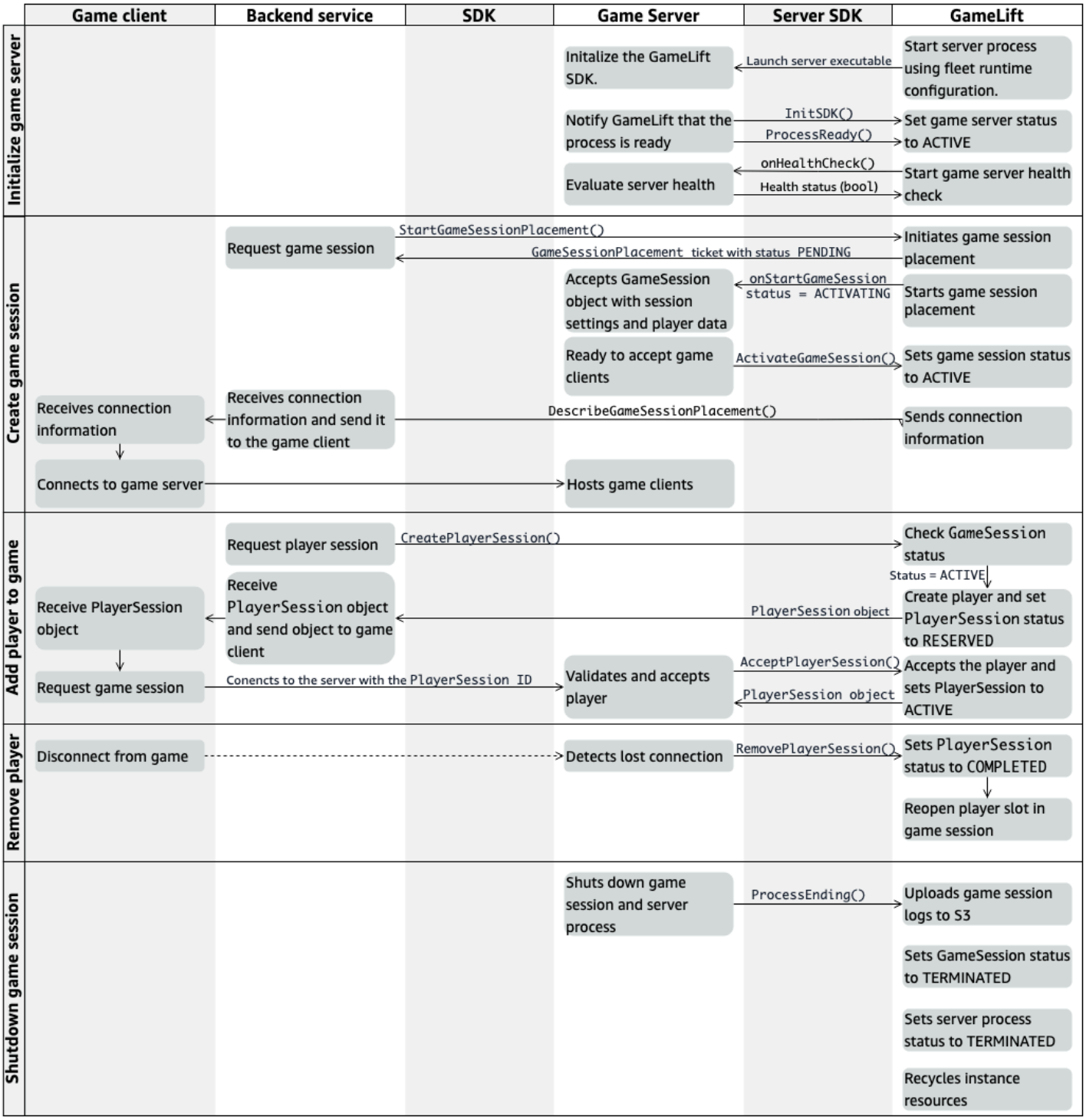
Topik

- Menguji integrasi Anda menggunakan Amazon GameLift Local

Interaksi server klien Amazon GameLift dan game

Topik ini menjelaskan interaksi antara klien game, layanan backend, server game, dan Amazon. GameLift

Diagram berikut menggambarkan interaksi antara klien game, layanan backend, Amazon GameLift SDK, server game EC2 terkelola, SDK GameLift server Amazon, dan Amazon. GameLift Untuk penjelasan rinci tentang interaksi yang ditampilkan, lihat bagian berikut di halaman ini.



Inisialisasi server game

Langkah-langkah berikut menjelaskan interaksi yang terjadi saat Anda menyiapkan server game untuk menyelenggarakan sesi game.

1. Amazon GameLift meluncurkan server yang dapat dieksekusi pada instans Amazon Elastic Compute Cloud (Amazon EC2).
2. Panggilan server game:

   a. `InitSDK()` untuk menginisialisasi server SDK.

   b. `ProcessReady()`untuk mengkomunikasikan kesiapan sesi permainan, informasi koneksi, dan lokasi file log sesi game.

   Proses server kemudian menunggu callback dari Amazon. GameLift
3. Amazon GameLift memperbarui status proses server `ACTIVE` untuk mengaktifkan penempatan sesi game.
4. Amazon GameLift mulai memanggil on`HealthCheck` callback dan terus memanggilnya secara berkala saat proses server aktif. Proses server dapat melaporkan sehat atau tidak sehat dalam satu menit.

Buat sesi game

Setelah Anda menginisialisasi server game Anda, interaksi berikut terjadi saat Anda membuat sesi game untuk meng-host pemain Anda.

1. Layanan backend memanggil operasi SDK. `StartGameSessionPlacement()`
2. Amazon GameLift membuat `GameSessionPlacement` tiket baru dengan status `PENDING` dan mengembalikannya ke layanan backend.
3. Layanan backend memperoleh status tiket penempatan dari antrian. Untuk informasi selengkapnya, lihat Atur notifikasi kejadian untuk penempatan sesi game.
4. Amazon GameLift memulai penempatan sesi game dengan memilih armada yang sesuai dan mencari proses server aktif dalam armada dengan sesi `0` game. Saat Amazon GameLift menemukan proses server, Amazon GameLift melakukan hal berikut:

   a. Membuat `GameSession` objek dengan pengaturan sesi permainan dan data pemain dari permintaan penempatan dengan `ACTIVATING` status.

   b. Memanggil callback on`StartGameSession` pada proses server. Amazon GameLift meneruskan informasi ke `GameSession` objek yang menunjukkan bahwa proses server dapat mengatur sesi permainan.

   c. Mengubah jumlah sesi game proses server menjadi1.

5. Proses server menjalankan fungsi on`StartGameSession` callback. Ketika proses server siap menerima koneksi pemain, ia memanggil `ActivateGameSession()` dan menunggu koneksi pemain.
6. Amazon GameLift memperbarui `GameSession` objek dengan informasi koneksi untuk proses server. (Informasi ini mencakup pengaturan port yang dilaporkan dengan`ProcessReady()`.) Amazon GameLift juga mengubah statusnya menjadi`ACTIVE`.
7. Layanan backend memanggil `DescribeGameSessionPlacement()` untuk mendeteksi status tiket yang diperbarui. Layanan backend kemudian menggunakan informasi koneksi untuk menghubungkan klien game ke proses server dan bergabung dengan sesi permainan.

Menambahkan pemain ke permainan

Urutan ini menjelaskan proses menambahkan pemain ke sesi game yang ada. Sesi pemain juga dapat diminta sebagai bagian dari permintaan penempatan sesi game.

1. Layanan backend memanggil operasi API klien `CreatePlayerSession()` dengan ID sesi game.
2. Amazon GameLift memeriksa status sesi permainan (harus`ACTIVE`), dan mencari slot pemain terbuka di sesi permainan. Jika slot tersedia, maka Amazon GameLift melakukan hal berikut:
    a. Menciptakan `PlayerSession` objek baru dan menetapkan status untuk`RESERVED`.
    b. Menanggapi permintaan layanan backend dengan objek. `PlayerSession`
3. Layanan backend menghubungkan klien game langsung ke proses server dengan ID sesi pemain.
4. Server memanggil operasi API server `AcceptPlayerSession()` untuk memvalidasi ID sesi pemain. Jika divalidasi, maka Amazon GameLift meneruskan `PlayerSession` objek ke proses server. Proses server bisa menerima atau menolak connection.
5. Amazon GameLift melakukan salah satu dari berikut ini:
    a. Jika koneksi diterima, maka Amazon GameLift menetapkan `PlayerSession` statusnya`ACTIVE`.
    b. Jika tidak ada respons yang diterima dalam 60 detik dari `CreatePlayerSession()` panggilan asli server backend, maka Amazon GameLift mengubah `PlayerSession` statusnya `TIMEDOUT` dan membuka kembali slot pemain di sesi permainan.

Menghapus pemain

Saat menghapus pemain dari sesi permainan untuk menciptakan ruang bagi pemain baru untuk bergabung, interaksi berikut terjadi.

1. Seorang pemain terputus dari permainan.
2. Server mendeteksi koneksi yang hilang dan memanggil operasi `RemovePlayerSession()` API server.
3. Amazon GameLift mengubah `PlayerSession` status `COMPLETED` dan membuka kembali slot pemain di sesi permainan.

Matikan sesi permainan

Urutan interaksi ini terjadi ketika proses server mematikan sesi permainan saat ini.

1. Server mematikan sesi permainan dan server.
2. Server memanggil `ProcessEnding()` ke AmazonGameLift.
3. Amazon GameLift melakukan hal berikut:
    a. Mengunggah log sesi game ke Amazon Simple Storage Service (Amazon S3).
    b. Mengubah `GameSession` status menjadi`TERMINATED`.
    c. Mengubah status proses server menjadi`TERMINATED`.
    d. Mendaur ulang sumber daya contoh.

Integrasikan server game Anda dengan Amazon GameLift

Setelah server game kustom Anda diterapkan dan berjalan di GameLift instans Amazon, server tersebut harus dapat berinteraksi dengan Amazon GameLift (dan berpotensi sumber daya lainnya). Bagian ini menjelaskan cara mengintegrasikan perangkat lunak server game Anda dengan AmazonGameLift.

> **Note**
>
> Instruksi ini mengasumsikan bahwa Anda telah membuat Akun AWS dan bahwa Anda memiliki proyek server game yang sudah ada.

Topik di bagian ini mendeskripsikan cara menangani tugas integrasi berikut:

- Buat komunikasi antara Amazon GameLift dan server game Anda.
- Buat 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 sedang mereka jalankan.

Topik

Tambahkan Amazon GameLift ke server game Anda

Server game khusus Anda harus berkomunikasi dengan AmazonGameLift, karena setiap proses server game harus dapat merespons peristiwa yang GameLift dimulai Amazon. Server game Anda juga harus terus GameLift menginformasikan Amazon tentang status proses server dan koneksi pemain. Untuk informasi selengkapnya tentang bagaimana server game, layanan backend, klien game, dan Amazon GameLift bekerja sama untuk mengelola hosting game, lihatInteraksi server klien Amazon GameLift dan game.

Untuk mempersiapkan server game Anda agar berinteraksi dengan AmazonGameLift, tambahkan Amazon GameLift Server SDK ke proyek server game Anda dan buat fungsionalitas yang dijelaskan dalam topik ini. Server SDK tersedia dalam beberapa bahasa. Untuk informasi selengkapnya tentang Amazon GameLift Server SDK, lihatDukungan pengembangan dengan Amazon GameLift.

Referensi API SDK Server:

- Referensi SDK 5.x GameLift server Amazon untuk C++
- Referensi SDK 5.x GameLift server Amazon untuk C # dan Unity
- Referensi SDK GameLift 5.x server Amazon Unreal Engine

Inisialisasi proses server

Tambahkan kode untuk menjalin komunikasi dengan Amazon GameLift dan melaporkan bahwa proses server siap untuk menjadi tuan rumah sesi permainan. Kode ini harus berjalan sebelum GameLift kode Amazon apa pun.

1. Inisialisasi klien Amazon GameLift API dengan meneleponInitSdk(). Untuk menginisialisasi proses server pada sumber daya GameLift Anywhere komputasi Amazon, panggil 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 hosting proses server game Anda.
   - ID GameLift armada yang berisi GameLift Anywhere komputasi Amazon Anda.
   - Token otorisasi yang dihasilkan oleh GameLift operasi GetComputeAuthToken Amazon.

   > **Note**
   >
   > Untuk menginisialisasi server game pada instans Amazon EC2 yang GameLift dikelola Amazon, buat konstruktor default InitSDK( )(C++) (C#) (Unreal) (C++) (C#) (Unreal. ServerParameters Amazon GameLift menyiapkan lingkungan komputasi dan secara otomatis terhubung ke Amazon GameLift untuk Anda.

2. Beri tahu Amazon GameLift bahwa proses server siap untuk menjadi tuan rumah sesi game. Panggilan ProcessReady( ) (C ++) (C #) (Unreal) (C ++) dengan informasi berikut. (Perhatikan bahwa Anda harus menelepon ProcessReady( ) hanya sekali per proses server).

   - Nomor port yang digunakan proses server. Layanan backend menyediakan nomor port dan alamat IP untuk klien game untuk terhubung ke proses server dan bergabung dengan sesi permainan.
   - Lokasi file, seperti log sesi game, yang ingin Anda pertahankan oleh AmazonGameLift. Proses server menghasilkan file-file ini selama sesi permainan. Mereka sementara disimpan pada contoh di mana proses server berjalan, dan mereka hilang ketika instance dimatikan. File apa pun yang Anda daftar diunggah ke AmazonGameLift. Anda dapat mengakses file-file ini melalui GameLiftkonsol Amazon atau dengan memanggil operasi Amazon GameLift API GetGameSessionLogUrl().

- Nama-nama fungsi callback yang GameLift dapat dipanggil Amazon ke proses server Anda. Server game Anda harus mengimplementasikan fungsi-fungsi ini. Untuk informasi lebih lanjut, lihat (C ++) (C #) (Unreal) (C ++)Unreal).
  - (Opsional) `onHealthCheck` - Amazon GameLift memanggil fungsi ini secara teratur untuk meminta laporan status kesehatan dari server.
  - `onStartGameSession`— Amazon GameLift memanggil fungsi ini sebagai respons terhadap permintaan klien CreateGameSession().
  - `onProcessTerminate`- Amazon GameLift memaksa proses server untuk berhenti, membiarkannya ditutup dengan anggun.
  - (Opsional) `onUpdateGameSession` — Amazon GameLift mengirimkan objek sesi game yang diperbarui ke server game atau memberikan pembaruan status pada permintaan isi ulang pertandingan. Fitur FlexMatchisi ulang memerlukan callback ini.

Anda juga dapat mengatur server game untuk mengakses AWS sumber daya yang Anda miliki atau kontrol dengan aman. Untuk informasi selengkapnya, lihat Berkomunikasi dengan sumber daya AWS lain dari armada.

(Opsional) Laporkan kesehatan proses server

Tambahkan kode ke server game Anda untuk mengimplementasikan fungsi `onHealthCheck()` callback. Amazon GameLift 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 tanggapi panggilan balik dalam waktu 60 detik. Jika Amazon GameLift tidak menerima respons pada waktu itu, Amazon 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 menganggap proses server menjadi sehat kecuali server tidak merespons.

Amazon GameLift 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 Amazon GameLift mungkin menutup proses dan memulai yang baru. Amazon GameLift 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 instans. Untuk mendapatkan lokasi file, panggil `GetComputeCertificate()`(C ++) (C #) (Unreal) (C ++)Unreal).

Mulai sesi game

Menambahkan kode untuk mengimplementasikan fungsionalitas callback `onStartGameSession`. Amazon GameLift memanggil callback ini untuk memulai sesi game di server.

`onStartGameSession`Fungsi ini mengambil GameSessionobjek sebagai parameter input. Objek ini mencakup informasi sesi permainan kunci, 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 berbasickan `GameSession`properti. Minimal, server game harus mengaitkan ID sesi game, referensi klien game mana saat menghubungkan ke proses server.
- Data proses game dan data pemain yang diperlukan. Data ini ada di `GameSession` objek.
- Beri tahu Amazon GameLift saat sesi permainan baru siap menerima pemain. Panggil operasi API server `ActivateGameSession()`(C ++) (C #) (Unreal) (C ++)Unreal). Menanggapi panggilan yang berhasil, Amazon GameLift mengubah status sesi game menjadi`ACTIVE`.

(Opsional) Validasi pemain baru

Jika Anda melacak status sesi pemain, tambahkan kode untuk memvalidasi pemain baru saat terhubung ke server game. Amazon GameLift melacak pemain saat ini dan slot sesi permainan yang tersedia.

Untuk validasi, klien game yang meminta akses ke sesi game harus menyertakan ID sesi pemain. Amazon GameLift secara otomatis menghasilkan ID ini ketika pemain meminta untuk bergabung dengan game menggunakan StartGameSessionPlacement() atau StartMatchmaking(). Sesi pemain kemudian cadangan slot terbuka dalam sesi permainan.

Ketika proses server game menerima permintaan koneksi klien game, ia memanggil `AcceptPlayerSession()` (C ++) (C #) (Unreal) (C ++) dengan ID sesi pemain. Sebagai tanggapan, Amazon GameLift memverifikasi bahwa ID sesi pemain sesuai dengan slot terbuka yang disediakan dalam sesi permainan. Setelah Amazon GameLift memvalidasi ID sesi pemain, proses server menerima koneksi. Pemain kemudian dapat bergabung dengan sesi permainan. Jika Amazon GameLift tidak memvalidasi ID sesi pemain, maka proses server akan menolak koneksi.

(Opsional) Laporkan akhir sesi pemain

Jika Anda melacak status sesi pemain, tambahkan kode untuk memberi tahu Amazon GameLift saat pemain meninggalkan sesi permainan. Kode ini harus berjalan setiap kali proses server mendeteksi connection jatuh. Amazon GameLift menggunakan notifikasi ini untuk melacak pemain saat ini dan slot yang tersedia di sesi permainan.

Untuk menangani koneksi yang terjatuh, dalam kode Anda, tambahkan panggilan ke operasi API server (C ++) `RemovePlayerSession()` (C #) (Unreal) (C ++) dengan ID sesi pemain yang sesuai.

Mengakhiri sesi game

Tambahkan kode ke urutan shutdown proses server untuk memberi tahu Amazon GameLift saat sesi game berakhir. Untuk mendaur ulang dan menyegarkan sumber daya hosting, Amazon GameLift mematikan proses server setelah sesi game selesai.

Pada awal kode shutdown proses server, panggil operasi API server (C ++) `ProcessEnding()` (C #) (Unreal) (C ++)Unreal). Panggilan ini memberi tahu Amazon GameLift bahwa proses server dimatikan. Amazon GameLift mengubah status sesi game dan status proses server menjadi`TERMINATED`. Setelah meneleponProcessEnding(), aman untuk proses untuk ditutup.

Menanggapi notifikasi shutdown proses server

Tambahkan kode untuk mematikan proses server sebagai respons terhadap pemberitahuan dari AmazonGameLift. Amazon GameLift mengirimkan notifikasi ini ketika proses server secara konsisten melaporkan tidak sehat, atau jika instans tempat proses server berjalan sedang dihentikan. Amazon GameLift dapat menghentikan instans sebagai bagian dari kejadian penurunan skala kapasitas, atau sebagai respons terhadap gangguan Instans Spot.

Untuk menangani pemberitahuan shutdown, lakukan perubahan berikut pada kode server game Anda:

- Menerapkan fungsionalitas panggilan balik onProcessTerminate(). fungsionalitas ini harus memanggil kode yang menutup proses server. Saat Amazon GameLift memanggil operasi ini,

interupsi Instans Spot memberikan pemberitahuan dua menit. Pemberitahuan ini memberikan waktu proses server untuk memutuskan pemain anggun, melestarikan data negara permainan, dan melakukan tugas-tugas pembersihan lainnya.

- Panggil operasi API server `GetTerminationTime()` (C ++) (C #) (Unreal) (C ++) shutdown server game Anda. Jika Amazon GameLift 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) (C ++)Unreal). Panggilan ini memberi tahu Amazon GameLift bahwa proses server dimatikan, dan Amazon GameLift kemudian mengubah status proses server menjadi. `TERMINATED` Setelah meneleponProcessEnding(), aman untuk proses untuk ditutup.

Berkomunikasi dengan sumber daya AWS lain dari armada

Saat membuat build server game untuk penyebaran di GameLift armada Amazon, Anda mungkin ingin aplikasi di build game Anda berkomunikasi secara langsung dan aman dengan AWS sumber daya lain yang Anda miliki. Karena Amazon GameLift mengelola armada hosting game Anda, Anda harus memberi Amazon akses GameLift terbatas ke sumber daya dan layanan ini.

Beberapa contoh skenario meliputi:

- Gunakan CloudWatch agen Amazon untuk mengumpulkan metrik, log, dan jejak dari armada dan armada EC2 yang dikelola 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 mendukung metode ini untuk membuat akses:

- Akses AWS sumber daya dengan peran IAM
- Mengakses sumber daya AWS dengan peering VPC

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 kredensil 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 kredensil
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 GameLift hak Amazon untuk menggunakan izin peran tersebut.

Untuk petunjuk tentang cara mengatur peran IAM, lihatMenyiapkan peran layanan IAM untuk Amazon GameLift. 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.

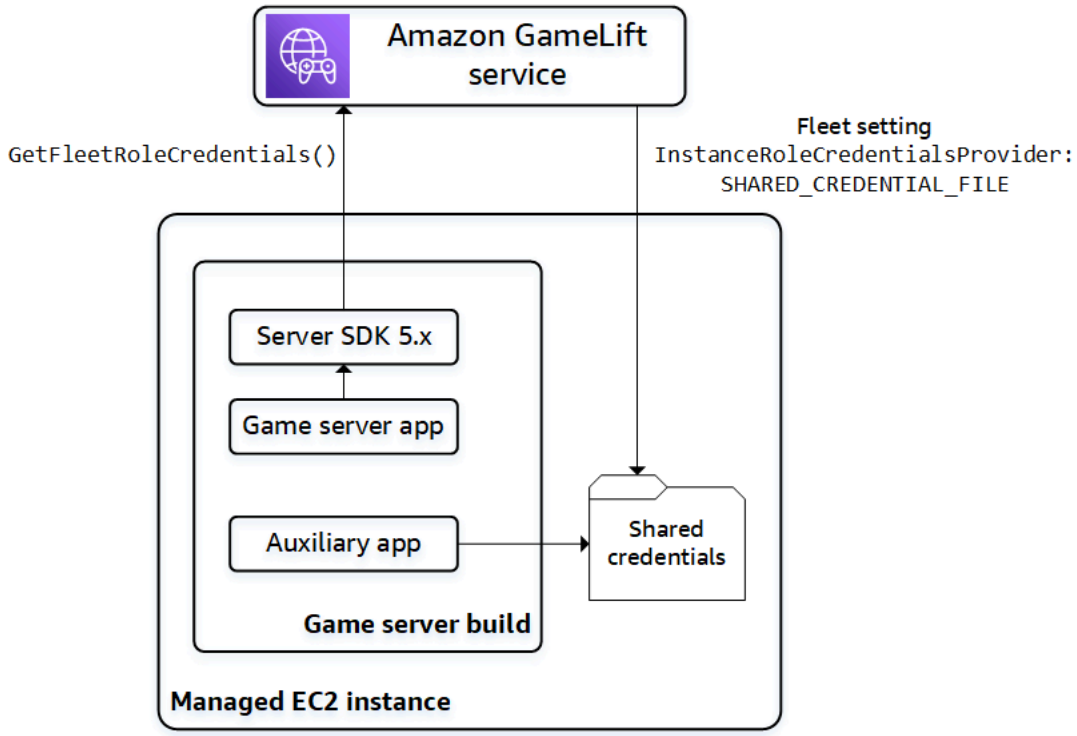
Memodifikasi aplikasi untuk memperoleh kredensil

Pada langkah ini, Anda mengonfigurasi aplikasi Anda untuk memperoleh kredensil keamanan untuk peran IAM dan menggunakannya saat berinteraksi dengan sumber daya Anda. AWS Lihat tabel berikut untuk menentukan cara memodifikasi aplikasi berdasarkan (1) jenis aplikasi, dan (2) versi SDK server yang digunakan game Anda untuk berkomunikasi dengan Amazon GameLift.

	Aplikasi server game	Aplikasi lainnya
Menggunakan server SDK versi 5.x	Panggil metode SDK server `GetFleetRoleCredentials()` dari kode server game Anda.	Tambahkan kode ke aplikasi untuk menarik kredensil dari file bersama pada instance armada.

	Aplikasi server game	Aplikasi lainnya
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 kredensil untuk peran IAM.



Panggilan **`GetFleetRoleCredentials()`** (server SDK 5.x)

Dalam kode server game Anda, yang seharusnya sudah terintegrasi dengan GameLift server Amazon SDK 5.x, panggil `GetFleetRoleCredentials` (C++) (C#) (Unreal) untuk mengambil satu set kredenal sementara. Saat kredensialnya kedaluwarsa, Anda dapat menyegarkannya dengan panggilan lain. `GetFleetRoleCredentials`

Gunakan kredenal 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 kredensil yang disimpan dalam file bersama. Amazon GameLift menghasilkan profil kredensil untuk setiap instance armada.

Kredensialnya tersedia untuk digunakan oleh semua aplikasi pada instance. Amazon GameLift terus menyegarkan kredensil sementara.

Anda harus mengonfigurasi armada untuk menghasilkan file kredensi bersama pada pembuatan armada.

Di setiap aplikasi yang perlu menggunakan file kredensi 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: Mengatur CloudWatch agen untuk mengumpulkan metrik untuk instans GameLift armada Amazon

Jika Anda ingin menggunakan CloudWatch agen Amazon untuk mengumpulkan metrik, log, dan jejak dari GameLift armada Amazon Anda, gunakan metode ini untuk mengotorisasi agen untuk 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 kredensil untuk berinteraksi dengan sumber daya Anda. AWS Aplikasi apa pun yang berjalan pada instance GameLift armada Amazon dengan server SDK 4 atau yang lebih lama 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 kredensil 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 kredenal 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 membuat file kredensial 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.

Mengakses sumber daya AWS dengan peering VPC

Anda dapat menggunakan peering Amazon Virtual Private Cloud (Amazon VPC) untuk berkomunikasi antara aplikasi yang berjalan pada GameLift instans Amazon dan sumber daya lainnya. AWS VPC adalah jaringan pribadi virtual yang Anda tentukan yang mencakup serangkaian sumber daya yang dikelola melalui Anda. Akun AWS Setiap GameLift armada Amazon memiliki VPC sendiri. Dengan VPC peering, Anda dapat membuat koneksi jaringan langsung antara VPC untuk armada Anda dan sumber daya lainnya. AWS

Amazon GameLift merampingkan 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

Integrasikan klien game Anda dengan Amazon GameLift

Topik di bagian ini menjelaskan GameLift fungsionalitas Amazon terkelola yang dapat Anda tambahkan ke layanan backend. Layanan backend menangani tugas-tugas berikut:

- Meminta informasi tentang sesi game aktif dari AmazonGameLift.

- 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 selengkapnya tentang cara klien game berinteraksi dengan Amazon GameLift dan server game yang berjalan di AmazonGameLift, lihatInteraksi server klien Amazon GameLift dan game.

Prasyarat

- Sesi Akun AWS.
- Build server game yang diunggah ke AmazonGameLift.
- Armada untuk hosting game Anda.

Topik

Tambahkan Amazon GameLift ke klien game Anda

Integrasikan Amazon GameLift 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 yang GameLift dihosting Amazon, lihat Panduan GameLift FlexMatchPengembang Amazon.

Siapkan Amazon GameLift di layanan backend

Tambahkan kode untuk menginisialisasi GameLift klien Amazon dan menyimpan pengaturan kunci. Kode ini harus berjalan sebelum kode apa pun bergantung pada Amazon GameLift.

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 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 GameLift klien Amazon. Gunakan Aws:GameLift::: GameLiftClient () (C ++) atau AmazonGameLiftClient() (C #) dengan konfigurasi klien default atau konfigurasi klien khusus.

3. Tambahkan mekanisme untuk menghasilkan pengenal unik bagi setiap pemain. Untuk informasi selengkapnya, lihat Hasilkan ID pemain.

4. Kumpulkan dan simpan informasi berikut:

   - Armada target - Banyak permintaan GameLift API Amazon 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 ke Amazon GameLift harus memberikan kredensi untuk yang Akun AWS menghosting game. Anda memperoleh kredensyal 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 API. Permintaan untuk mengambil peran mengembalikan satu set kredensyal keamanan sementara. Untuk informasi selengkapnya, lihat Beralih ke peran IAM (AWS API) di Panduan Pengguna IAM.
     - Jika Anda memiliki kredensi keamanan jangka panjang, konfigurasikan kode Anda untuk menemukan dan menggunakan kredensi yang disimpan. Lihat Mengautentikasi menggunakan kredensi jangka panjang di Panduan Referensi AWS SDK dan Alat. Untuk informasi tentang menyimpan kredensional, lihat referensi AWS API untuk (C++) dan (.NET).

- Jika Anda memiliki kredenal keamanan sementara, tambahkan kode untuk menyegarkan kredensional secara teratur menggunakan AWS Security Token Service (AWS STS), seperti yang dijelaskan dalam Menggunakan kredenal keamanan sementara dengan AWS SDK di Panduan Pengguna IAM. Kode harus meminta kredensyal 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 SearchGameSessionsuntuk mendapatkan informasi tentang sesi permainan tertentu, semua sesi aktif, atau sesi yang memenuhi serangkaian kriteria pencarian. Panggilan ini mengembalikan GameSessionobjek 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: Evaluasi`CreationTime`.
- 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 StartGameSessionPlacementuntuk menempatkan permintaan untuk sesi permainan baru dalam antrian. Untuk menggunakan operasi ini, buat antrian. Ini menentukan di mana Amazon GameLift menempatkan sesi permainan baru. Untuk informasi lebih lanjut tentang antrian dan cara menggunakannya, lihat. Menyiapkan GameLift antrean Amazon untuk penempatan sesi game

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, maka Amazon GameLift menggunakan informasi ini untuk menempatkan sesi permainan baru pada armada yang memberikan pengalaman gameplay yang ideal bagi para pemain.

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 CreateGameSessionuntuk 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 membuat sesi permainan baru dan mengembalikan GameSessionobjek, Anda dapat bergabung dengan pemain ke sana.

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 GameLift sumber daya Amazon 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, lihatBagaimana pemain terhubung ke game.

   Ada dua cara untuk membuat sesi pemain baru:

   - Gunakan StartGameSessionPlacementuntuk memesan slot untuk satu atau lebih pemain di sesi permainan baru.
   - Cadangan slot pemain untuk satu atau lebih pemain yang menggunakan CreatePlayerSessionatau CreatePlayerSessionsdengan ID sesi permainan.

   Amazon GameLift pertama kali memverifikasi bahwa sesi permainan menerima pemain baru dan memiliki slot pemain yang tersedia. Jika berhasil, Amazon GameLift mencadangkan slot untuk pemain, membuat sesi pemain baru, dan mengembalikan PlayerSessionobjek. 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 ID pemain.

   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 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 membuat koneksi, klien game dan proses server berkomunikasi langsung tanpa melibatkan Amazon GameLift. Server memelihara komunikasi dengan Amazon GameLift 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 dipesan 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 game baru dari Amazon GameLift 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 tersebut, Amazon GameLift membuat `GameSession` objek yang berisi properti game yang ditentukan. Amazon GameLift kemudian menginstruksikan server game yang tersedia untuk memulai sesi permainan baru dan melewati `GameSession` objek. Server game memulai sesi permainan dengan `Difficulty` a`Easy`.

Pelajari selengkapnya

- GameProperty tipe data
- SearchGameSessions() contoh
- UpdateGameSession() GameProperties parameter

Hasilkan ID pemain

Amazon GameLift menggunakan sesi pemain untuk mewakili pemain yang terhubung ke sesi permainan. Amazon GameLift membuat sesi pemain setiap kali pemain terhubung ke sesi permainan menggunakan klien game yang terintegrasi dengan AmazonGameLift. Ketika seorang pemain meninggalkan permainan, sesi pemain berakhir. Amazon GameLift tidak menggunakan kembali sesi pemain.

Contoh kode berikut secara acak menghasilkan ID pemain unik:

```
bool includeBrackets = false;
bool includeDashes = true;
string playerId = AZ::Uuid::CreateRandom().ToString<string>(includeBrackets,
 includeDashes);
```

Untuk informasi selengkapnya tentang sesi pemain, lihatMelihat data pada sesi game dan pemain.

Mesin game dan Amazon GameLift

Anda dapat menggunakan GameLift layanan Amazon yang dikelola 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 selengkapnya tentang Amazon GameLift SDK yang tersedia, platform pengembangan yang didukung, dan sistem operasi, lihat Dukungan pengembangan dengan Amazon GameLift server game.

Selain informasi khusus mesin yang disediakan dalam topik ini, temukan bantuan tambahan untuk mengintegrasikan Amazon GameLift ke server game, klien, dan layanan Anda dalam topik berikut:

- Peta jalan hosting yang GameLift dikelola Amazon- Alur kerja enam langkah untuk berhasil mengintegrasikan Amazon GameLift ke dalam game Anda dan menyiapkan sumber daya hosting.
- Tambahkan Amazon GameLift ke server game Anda- Instruksi terperinci tentang mengintegrasikan Amazon GameLift ke server game.
- Tambahkan Amazon GameLift 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 di Amazon GameLift menggunakan Amazon GameLift Server SDK untuk C++. Lihat Tambahkan Amazon GameLift ke server game Anda untuk mendapatkan bantuan dengan mengintegrasikan fungsionalitas yang diperlukan ke server game Anda.

Klien dan layanan Game

Aktifkan klien game dan/atau layanan game Anda untuk berinteraksi dengan GameLift layanan Amazon, seperti menemukan sesi permainan yang tersedia atau membuat yang baru, dan menambahkan pemain ke game. fungsionalitas klien inti disediakan diSDK AWS for C++. Untuk mengintegrasikan Amazon GameLift ke dalam proyek game O3DE Anda, lihat dan. Tambahkan Amazon GameLift ke klien dan server game O3DE Tambahkan Amazon GameLift ke klien game Anda

Unreal Engine

Server Game

Siapkan server game Anda untuk hosting di Amazon GameLift dengan menambahkan Amazon GameLift Server SDK for Unreal Engine ke proyek Anda dan menerapkan fungsionalitas server yang diperlukan. Untuk bantuan menyiapkan plugin Unreal Engine dan menambahkan GameLift kode Amazon, lihatIntegrasikan Amazon GameLift ke dalam proyek Unreal Engine.

Klien dan layanan Game

Aktifkan klien game dan/atau layanan game Anda untuk berinteraksi dengan GameLift layanan Amazon, seperti menemukan sesi permainan yang tersedia atau membuat yang baru, dan menambahkan pemain ke game. fungsionalitas klien inti disediakan diSDK AWS for C++. Untuk mengintegrasikan Amazon GameLift ke dalam proyek game Unreal Engine Anda, lihat. Tambahkan Amazon GameLift ke klien game Anda

Unity

Server game

Siapkan server game Anda untuk hosting di Amazon GameLift dengan menambahkan Amazon GameLift Server SDK untuk C # ke proyek Anda dan menerapkan fungsionalitas server yang

diperlukan. Untuk bantuan menyiapkan Unity dan menambahkan GameLift kode Amazon, lihatIntegrasikan Amazon GameLift ke dalam proyek Unity.

Klien dan layanan Game

Aktifkan klien game dan/atau layanan game Anda untuk berinteraksi dengan GameLift layanan Amazon, seperti menemukan sesi permainan yang tersedia atau membuat yang baru, dan menambahkan pemain ke game. fungsionalitas klien inti disediakan di AWS SDK for .NET. Untuk mengintegrasikan Amazon GameLift ke dalam proyek game Unity Anda, lihatTambahkan Amazon GameLift ke klien game Anda.

Mesin lainnya

Untuk daftar lengkap Amazon GameLift SDK yang tersedia untuk server game dan klien, lihatthe section called “Dukungan pengembangan”.

Tambahkan Amazon GameLift 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 kerja modular yang dapat Anda modifikasi dan perluas dengan alat pengembangan pilihan Anda.

Kerangka modular menggunakan Gems yang berisi pustaka dengan antarmuka standar dan aset. Pilih Gems Anda sendiri untuk memilih fungsi apa yang akan ditambahkan berdasarkan kebutuhan Anda.

Amazon GameLift Gem menyediakan fitur-fitur berikut:

GameLiftIntegrasi Amazon

Kerangka kerja untuk memperluas lapisan jaringan O3DE dan membiarkan Multiplayer Gem bekerja dengan solusi server GameLift khusus Amazon. Gem menyediakan integrasi dengan SDK GameLiftserver Amazon dan klien SDK (untuk memanggil layanan Amazon GameLift itu sendiri). AWS

Manajemen build dan 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.

Pengaturan Amazon GameLift Gem

Ikuti prosedur di bagian ini untuk mengatur Amazon GameLift Gem di O3DE.

Prasyarat

- Siapkan AWS akun Anda untuk AmazonGameLift. Untuk informasi selengkapnya, lihat Siapkan Akun AWS.
- Mengatur AWS kredensi untuk O3DE. Untuk informasi selengkapnya, lihat Mengkonfigurasi AWS Kredensial.
- Mengatur AWS CLI danAWS CDK. Untuk informasi lebih lanjut, AWS Command Line Interfacedan AWS Cloud Development Kit (AWS CDK).

Aktifkan Amazon GameLift Gem dan dependensinya

1. Buka Manajer Proyek.
2. Buka menu di bawah proyek Anda dan pilih Edit Pengaturan Proyek... .
3. Pilih Konfigurasikan Permata.
4. Aktifkan Amazon GameLift Gem dan Permata dependen berikut:
   - AWSCore Gem - Menyediakan kerangka kerja untuk digunakan Layanan AWS dalam O3DE.
   - Permata Multiplayer - Menyediakan fungsionalitas multipemain dengan memperluas kerangka jaringan.

Sertakan pustaka statis Amazon GameLift Gem

1. Sertakan `Gem::AWSGameLift.Server.Static` sebagai `BUILD_DEPENDENCIES` untuk target server proyek Anda.

```
ly_add_target(
    NAME YourProject.Server.Static STATIC
    ...
    BUILD DEPENDCIES
        PUBLIC
            ...
        PRIVATE
            ...
            Gem::AWSGameLift.Server.Static
```

```
)
```

2. Setel `AWSGameLiftService` ke diperlukan untuk komponen sistem server proyek Anda.

```
void
 YourProjectServerSystemComponent::GetRequiredServices(AZ::ComponentDescriptor::DependencyA
 required)
{
    ...
    required.push_back(AZ_CRC_CE("AWSGameLiftServerService"));
    ...
}
```

3. (Opsional) Untuk membuat permintaan GameLift layanan Amazon di C ++, sertakan `Gem::AWSGameLift.Client.Static` dalam `BUILD_DEPENDENCIES` 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 server khusus Anda

Kelola sesi permainan dalam game Anda dan server game khusus dengan Integrasi Manajemen Sesi. Untuk mendukungFlexMatch, lihat FlexMatchIntegrasi.

Integrasikan Amazon GameLift ke dalam proyek Unreal Engine

Topik ini menjelaskan cara menyiapkan plugin SDK server Amazon GameLift C ++ untuk Unreal Engine dan mengintegrasikannya ke dalam proyek game Anda.

Sumber daya tambahan:

- Plugin SDK server untuk situs unduhan Unreal

- Referensi SDK GameLift 5.x server Amazon Unreal Engine
- the section called “Dukungan pengembangan”

Prasyarat

Sebelum Anda memproses, pastikan Anda telah meninjau prasyarat berikut:

Prasyarat

- Komputer yang mampu menjalankan Unreal Engine. Untuk informasi selengkapnya tentang persyaratan Unreal Engine, lihat dokumentasi Spesifikasi Perangkat Keras dan Perangkat Lunak Unreal Engine.
- Microsoft Visual Studio 2019 atau versi yang lebih baru.
- CMake versi 3.1 atau yang lebih baru.
- Python versi 3.6 atau lebih baru.
- Klien Git tersedia di PATH.
- Akun game Epic. Mendaftar untuk mendapatkan akun di situs web resmi Unreal Engine.
- GitHub Akun yang terkait dengan akun Unreal Engine Anda. Untuk informasi lebih lanjut, lihat Mengakses kode sumber Unreal Engine GitHub di situs web Unreal Engine.

> **Note**
>
> Amazon GameLift saat ini mendukung versi Unreal Engine berikut:
>
> - 4.22
> - 4.23
> - 4.24
> - 4.25
> - 4.26
> - 4.27
> - 5.1.0
> - 5.1.1

Bangun Unreal Engine dari sumber

Versi standar editor Unreal Engine, diunduh melalui peluncur Epic, hanya mengizinkan pembuatan aplikasi klien Unreal. Untuk membangun aplikasi server Unreal, Anda perlu mengunduh dan membangun Unreal Engine dari sumber, menggunakan repo Unreal Engine Github. Untuk informasi lebih lanjut, lihat tutorial Building Unreal Engine from Source di situs web dokumentasi Unreal Engine.

> **Note**
>
> Jika Anda belum melakukannya, ikuti petunjuk di Mengakses kode sumber Unreal Engine untuk menautkan GitHub akun Anda GitHub ke akun Epic Games Anda.

Untuk mengkloning sumber Unreal Engine ke lingkungan pengembangan Anda

1. Kloning sumber Unreal Engine ke lingkungan pengembangan Anda di cabang pilihan Anda.

   ```
   git clone https://github.com/EpicGames/UnrealEngine.git
   ```

2. Lihat tag versi yang Anda gunakan untuk mengembangkan game Anda. Misalnya, contoh berikut memeriksa Unreal Engine versi 5.1.1:

   ```
   git checkout tags/5.1.1-release -b 5.1.1-release
   ```

3. Arahkan ke folder root dari repositori lokal. Saat Anda berada di folder root, jalankan file berikut:`Setup.bat`.
4. Saat berada di folder root, jalankan juga file:`GenerateProjectFiles.bat`.
5. Setelah menjalankan file dari langkah sebelumnya, file solusi Unreal Engine`UE5.sln`, dibuat. Buka Visual Studio, dan di editor Visual Studio buka `UE5.sln` file.
6. Di Visual Studio, buka menu View dan pilih opsi Solution Explorer. Ini membuka menu konteks dari node proyek Unreal. Di jendela Solution Explorer, klik kanan `UE5.sln` file (dapat dicantumkan sebagai hanya`UE5`), lalu pilih Build untuk membangun proyek Unreal dengan target Development Editor Win64.

   > **Note**
   >
   > Pembangunannya bisa memakan waktu lebih dari satu jam untuk menyelesaikannya.

Setelah build selesai, Anda siap untuk membuka Unreal Development Editor dan membuat atau mengimpor proyek.

Konfigurasikan proyek Unreal Anda untuk plugin

Ikuti langkah-langkah ini untuk menyiapkan plugin SDK GameLift server Amazon untuk Unreal Engine untuk proyek server game Anda.

Untuk mengkonfigurasi proyek Anda untuk plugin

1. Dengan Visual Studio terbuka, navigasikan ke panel Solution Explorer dan pilih UE5 file untuk membuka menu konteks untuk proyek Unreal. Dalam menu konteks, pilih opsi Set as Startup Project.
2. Di bagian atas jendela Visual Studio Anda, pilih Mulai Debugging (panah hijau).

   Tindakan ini memulai instance Unreal Editor buatan sumber baru Anda. Untuk informasi selengkapnya tentang penggunaan Unreal Editor, lihat Unreal Editor Interface di situs web dokumentasi Unreal Engine.
3. Tutup jendela Visual Studio yang Anda buka, karena Unreal Editor membuka jendela Visual Studio lain yang berisi proyek Unreal dan proyek game Anda.
4. Di editor Unreal, lakukan salah satu hal berikut:
   - Pilih proyek Unreal yang sudah ada yang ingin Anda integrasikan dengan Amazon GameLift.
   - Buat proyek baru. Untuk bereksperimen dengan GameLift plugin Amazon untuk Unreal, coba gunakan template Orang Ketiga Unreal engine. Untuk informasi selengkapnya tentang template ini, lihat template Orang Ketiga di situs web dokumentasi Unreal Engine.

     Atau, konfigurasikan proyek baru dengan pengaturan berikut:

     - C++
     - Dengan konten starter
     - Desktop
     - Sebuah nama proyek. Dalam contoh dalam topik ini, kami menamai proyek kami`GameLiftUnrealApp`.
5. Di Solution Explorer Visual Studio, arahkan ke lokasi proyek Unreal Anda. Di `Source` folder Unreal, temukan file bernama*`Your-application-name`*`.Target.cs`.

   Misalnya: `GameLiftUnrealApp.Target.cs`.

6. Buat salinan file ini dan beri nama salinannya:*Your-application-name*`Server.Target.cs`.
7. Buka file baru dan buat perubahan berikut:
   - Ubah `class` dan `constructor` untuk mencocokkan nama file.
   - Ubah `Type` dari `TargetType.Game` ke`TargetType.Server`.
   - File akhir akan terlihat seperti contoh berikut:

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

Proyek Anda sekarang dikonfigurasi untuk menerima plugin SDK GameLift server Amazon.

Tugas selanjutnya adalah membangun pustaka SDK server C ++ untuk Unreal sehingga Anda dapat mengimpornya ke proyek Anda.

Untuk membangun pustaka SDK server C ++ untuk Unreal

1. Unduh plugin SDK server Amazon GameLift C ++ untuk Unreal.

> **Note**
>
> Menempatkan SDK di direktori unduhan default dapat mengakibatkan kegagalan build karena jalur melebihi batas 260 karakter. Sebagai contoh: `C:\Users\Administrator\Downloads\GameLift-SDK-Release-06_15_2023\GameLift-Cpp-ServerSDK-5.0.4`
> Kami menyarankan Anda memindahkan SDK ke direktori lain, misalnya`C:\GameLift-Cpp-ServerSDK-5.0.4`.

2. Unduh dan instal OpenSSL. Untuk informasi lebih lanjut tentang mengunduh OpenSSL, baca dokumentasi build dan install Github OpenSSL.

Untuk informasi selengkapnya, baca dokumentasi OpenSSL Notes untuk platform Windows.

> **Note**
>
> Versi OpenSSL yang Anda gunakan untuk membangun SDK server GameLift Amazon harus sesuai dengan versi OpenSSL yang digunakan oleh Unreal untuk mengemas server game Anda. Anda dapat menemukan informasi versi di direktori `...Engine\Source\ThirdParty\OpenSSL` instalasi Unreal.

3. Dengan pustaka yang diunduh, buat pustaka SDK server C ++ untuk Unreal Engine.

   Di `GameLift-Cpp-ServerSDK-<version>` direktori di SDK yang diunduh, kompilasi dengan `-DBUILD_FOR_UNREAL=1` parameter dan buat SDK server. Contoh berikut menunjukkan cara mengkompilasi menggunakan`cmake`.

   Jalankan perintah berikut di terminal Anda:

   ```
   mkdir cmake-build
   cmake.exe -G "Visual Studio 17 2022" -DCMAKE_BUILD_TYPE=Release -S . -B ./cmake-
   build -DBUILD_FOR_UNREAL=1 -A x64
   cmake.exe --build ./cmake-build --target ALL_BUILD --config Release
   ```

   Build Windows membuat file biner berikut di `out\gamelift-server-sdk\Release` folder:

   - `cmake-build\prefix\bin\aws-cpp-sdk-gamelift-server.dll`
   - `cmake-build\prefix\bin\aws-cpp-sdk-gamelift-server.lib`

   Salin dua file pustaka ke `ThirdParty\GameLiftServerSDK\Win64` folder dalam paket plugin Amazon GameLift Unreal Engine.

Gunakan prosedur berikut untuk mengimpor GameLift plugin Amazon ke proyek contoh Anda.

Impor GameLift plugin Amazon

1. Temukan `GameLiftServerSDK` folder yang Anda ekstrak dari plugin di prosedur sebelumnya.
2. Temukan `Plugins` di folder root proyek game Anda. (Jika folder tidak ada, maka buatlah di sana.)

3. Salin `GameLiftServerSDK` folder ke dalam`Plugins`.

   Ini akan memungkinkan proyek Unreal untuk melihat plugin.

4. Tambahkan plugin SDK GameLift server Amazon ke `.uproject` file game.

   Dalam contoh, aplikasi dipanggil`GameLiftUnrealApp`, jadi file akan`GameLiftUnrealApp.uproject`.

5. Edit `.uproject` file untuk menambahkan plugin ke proyek game Anda.

```
"Plugins": [
    {
        "Name": "GameLiftServerSDK",
        "Enabled": true
    }
]
```

6. Pastikan game ModuleRules mengambil ketergantungan pada plugin. Buka `.Build.cs` file dan tambahkan dependensi Amazon GameLiftServer SDK. File ini ditemukan di bawah*Your-application-name*/Source/*/Your-application-name/*.

   Misalnya, filepath tutorial adalah. `../GameLiftUnrealApp/Source/GameLiftUnrealApp/GameLiftUnrealApp.Build.cs`

7. Tambahkan `"GameLiftServerSDK"` ke akhir daftar`PublicDependencyModuleNames`.

```
using UnrealBuildTool;
using System.Collections.Generic;
public class GameLiftUnrealApp : ModuleRules
  {
    public GameLiftUnrealApp(TargetInfo Target)
    {
      PublicDependencyModuleNames.AddRange(new string[] { "Core", "CoreUObject",
 "Engine", "InputCore", "GameLiftServerSDK" });
      bEnableExceptions = true;
    }
  }
```

Plugin sekarang harus berfungsi untuk aplikasi Anda. Lanjutkan dengan bagian selanjutnya untuk mengintegrasikan GameLift fungsionalitas Amazon ke dalam game Anda.

Tambahkan kode GameLift server Amazon ke proyek Unreal Anda

Anda telah mengonfigurasi dan menyiapkan lingkungan Unreal Engine Anda, dan sekarang Anda dapat mengintegrasikan server game dengan Amazon GameLift. Kode yang disajikan dalam topik ini membuat panggilan yang diperlukan ke GameLift layanan Amazon. Ini juga mengimplementasikan satu set fungsi callback yang merespons permintaan dari layanan Amazon GameLift . 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. Referensi SDK GameLift server Amazon untuk Unreal Engine

Untuk menginisialisasi server game dengan Amazon GameLift, gunakan prosedur berikut.

> **Note**
>
> Kode GameLift khusus Amazon yang disediakan di bagian berikut bergantung pada penggunaan flag `WITH_GAMELIFT` preprocessor. Bendera ini benar hanya jika kedua kondisi ini terpenuhi:
>
> - `Target.Type == TargetRules.TargetType.Server`
> - Plugin menemukan binari SDK GameLift server Amazon.
>
> Ini memastikan bahwa hanya build Unreal Server yang menjalankan API backend GameLift Amazon. Ini juga memungkinkan Anda untuk menulis kode yang akan dijalankan dengan benar untuk semua target Unreal yang berbeda yang mungkin dihasilkan game Anda.

Integrasikan server game dengan Amazon GameLift

1. Di Visual Studio, buka `.sln` file untuk aplikasi Anda. Sebagai contoh kita, file `GameLiftUnrealApp.sln` tersebut ditemukan di folder root.
2. Dengan solusi terbuka, cari *`Your-application-name`*`GameMode.h` file aplikasi Anda. Contoh:`GameLiftUnrealAppGameMode.h`.
3. Ubah file header agar sejajar dengan kode contoh berikut. Pastikan untuk mengganti "GameLiftUnrealApp" dengan nama aplikasi Anda sendiri.

```
#pragma once

#include "CoreMinimal.h"
```

```
#include "GameFramework/GameModeBase.h"
#include "GameLiftServerSDK.h"
#include "GameLiftUnrealAppGameMode.generated.h"

DECLARE_LOG_CATEGORY_EXTERN(GameServerLog, Log, All);

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

public:
    AGameLiftUnrealAppGameMode();

protected:
    virtual void BeginPlay() override;

private:
    // Process Parameters needs to remain in scope for the lifetime of the app
    FProcessParameters m_params;

    void InitGameLift();
};
```

4. Buka file *Your-application-name*GameMode.cpp file sumber terkait. Dalam Contoh kami:GameLiftUnrealAppGameMode.cpp. dan ubah kode untuk menyelaraskan dengan kode contoh berikut. Pastikan untuk mengganti "GameLiftUnrealApp" dengan nama aplikasi Anda sendiri.

   Contoh ini menunjukkan cara menambahkan semua elemen yang diperlukan untuk integrasi dengan Amazon GameLift, seperti yang dijelaskan dalam Tambahkan Amazon GameLift ke server game Anda. Hal ini mencakup:

   - Menginisialisasi klien GameLift API Amazon.
   - Menerapkan fungsi callback untuk menanggapi permintaan dari GameLift layanan Amazon, termasukOnStartGameSession,OnProcessTerminate, danonHealthCheck.
   - Memanggil ProcessReady () dengan port yang ditunjuk untuk memberi tahu Amazon GameLiftservice saat siap menyelenggarakan sesi game.

```
#include "GameLiftUnrealAppGameMode.h"
```

```
#include "GameLiftUnrealAppCharacter.h"

#include "UObject/ConstructorHelpers.h"

DEFINE_LOG_CATEGORY(GameServerLog);

AGameLiftUnrealAppGameMode::AGameLiftUnrealAppGameMode()
{
    // 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;
    }
}

void AGameLiftUnrealAppGameMode::BeginPlay()
{
#if WITH_GAMELIFT
    InitGameLift();
#endif
}

void AGameLiftUnrealAppGameMode::InitGameLift()
{
    UE_LOG(GameServerLog, Log, TEXT("Initializing the GameLift Server"));

    //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 serverParameters;

    //AuthToken returned from the "aws gamelift get-compute-auth-token" API. Note
 this will expire and require a new call to the API after 15 minutes.
    if (FParse::Value(FCommandLine::Get(), TEXT("-authtoken="),
 serverParameters.m_authToken))
    {
        UE_LOG(GameServerLog, Log, TEXT("AUTH_TOKEN: %s"),
 *serverParameters.m_authToken)
    }
```

```
   //The Host/compute-name of the GameLift Anywhere instance.
   if (FParse::Value(FCommandLine::Get(), TEXT("-hostid="),
serverParameters.m_hostId))
   {
       UE_LOG(GameServerLog, Log, TEXT("HOST_ID: %s"), *serverParameters.m_hostId)
   }

   //The Anywhere Fleet ID.
   if (FParse::Value(FCommandLine::Get(), TEXT("-fleetid="),
serverParameters.m_fleetId))
   {
       UE_LOG(GameServerLog, Log, TEXT("FLEET_ID: %s"),
*serverParameters.m_fleetId)
   }

   //The WebSocket URL (GameLiftServiceSdkEndpoint).
   if (FParse::Value(FCommandLine::Get(), TEXT("-websocketurl="),
serverParameters.m_webSocketUrl))
   {
       UE_LOG(GameServerLog, Log, TEXT("WEBSOCKET_URL: %s"),
*serverParameters.m_webSocketUrl)
   }

   //The PID of the running process
   serverParameters.m_processId = FString::Printf(TEXT("%d"),
GetCurrentProcessId());
   UE_LOG(GameServerLog, Log, TEXT("PID: %s"), *serverParameters.m_processId);

   //InitSDK establishes a local connection with GameLift's agent to enable
further communication.
   //Use InitSDK(serverParameters) for a GameLift Anywhere fleet.
   //Use InitSDK() for a GameLift managed EC2 fleet.
   gameLiftSdkModule->InitSDK(serverParameters);

   //Implement callback function onStartGameSession
   //GameLift sends a game session activation request to the game server
   //and passes a game session object with 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().
   auto onGameSession = [=](Aws::GameLift::Server::Model::GameSession gameSession)
   {
       FString gameSessionId = FString(gameSession.GetGameSessionId());
```

```
        UE_LOG(GameServerLog, Log, TEXT("GameSession Initializing: %s"),
*gameSessionId);
        gameLiftSdkModule->ActivateGameSession();
    };

    m_params.OnStartGameSession.BindLambda(onGameSession);

    //Implement callback function OnProcessTerminate
    //GameLift 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 GameLift it is shutting down.
    auto onProcessTerminate = [=]()
    {
        UE_LOG(GameServerLog, Log, TEXT("Game Server Process is terminating"));
        gameLiftSdkModule->ProcessEnding();
    };

    m_params.OnTerminate.BindLambda(onProcessTerminate);

    //Implement callback function OnHealthCheck
    //GameLift 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 GameLift records 'false'.
    //In this example, the game server always reports healthy.
    auto onHealthCheck = []()
    {
        UE_LOG(GameServerLog, Log, TEXT("Performing Health Check"));
        return true;
    };

    m_params.OnHealthCheck.BindLambda(onHealthCheck);

    //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.
    m_params.port = 7777;

    //Here, the game server tells GameLift where to find game session log files.
    //At the end of a game session, GameLift uploads everything in the specified
    //location and stores it in the cloud for access later.
```

```
    TArray<FString> logfiles;
    logfiles.Add(TEXT("GameLift426Test/Saved/Logs/GameLift426Test.log"));
    m_params.logParameters = logfiles;

    //The game server calls ProcessReady() to tell GameLift it's ready to host game
 sessions.
    UE_LOG(GameServerLog, Log, TEXT("Calling Process Ready"));
    gameLiftSdkModule->ProcessReady(m_params);
}
```

5. Membangun proyek game untuk kedua jenis target berikut: Development Editor dan Development Server.

> **Note**
>
> Anda tidak perlu membangun kembali solusinya. Sebagai gantinya, buat hanya proyek di bawah `Games` folder yang cocok dengan nama aplikasi Anda. Jika tidak, Visual Studio membangun kembali seluruh proyek UE5, yang mungkin memakan waktu hingga satu jam.

6. Setelah kedua build selesai, tutup Visual Studio dan buka `.uproject` file proyek Anda untuk membukanya di Unreal Editor.
7. Di Unreal Editor, paketkan server build game Anda. Untuk memilih target, buka Platform, Windows dan pilih ***`Your-application-name Server`***.
8. Untuk memulai proses membangun aplikasi server, buka Platform, Windows dan pilih Package Project. Saat build selesai, Anda harus memiliki executable. Dalam kasus contoh kita, nama file adalah`GameLiftUnrealAppServer.exe`.
9. Membangun aplikasi server di Unreal Editor menghasilkan dua executable. Satu terletak di root folder build game dan bertindak sebagai pembungkus untuk server yang sebenarnya dapat dieksekusi.

   Saat membuat GameLift armada Amazon dengan build server Anda, sebaiknya Anda meneruskan server yang dapat dieksekusi sebagai jalur peluncuran konfigurasi runtime. Misalnya, di folder build game Anda, Anda mungkin memiliki `GameLiftFPS.exe` file di root dan file lainnya di`\GameLiftFPS\Binaries\Win64\GameLiftFPSServer.exe`. Saat membuat armada, kami sarankan Anda menggunakan `C:\GameLiftFPS\Binaries\Win64\GameLiftFPSServer.exe` sebagai jalur peluncuran konfigurasi runtime.

10. Pastikan untuk membuka port UDP yang diperlukan pada GameLift armada Amazon, sehingga server game dapat berkomunikasi dengan klien game. Secara default, Unreal Engine menggunakan port7777. Untuk informasi selengkapnya, lihat UpdateFleetPortSettingsdi panduan referensi API GameLift layanan Amazon.
11. Buat `install.bat` file untuk build game Anda. Skrip penginstalan ini berjalan setiap kali game build dikerahkan ke GameLift armada Amazon. Berikut adalah contoh `install.bat` file:

    ```
    VC_redist.x64.exe /q
    UE5PrereqSetup_x64.exe /q
    ```

    Untuk beberapa versi Unreal Engine, `install.bat` seharusnya

    ```
    VC_redist.x64.exe /q
    UEPrereqSetup_x64.exe /q
    ```

    > **Note**
    >
    > Jalur file ke `<>PrereqSetup_x64.exe` file tersebut adalah`Engine\Extras\Redist\en-us`.

12. Sekarang Anda dapat mengemas dan mengunggah build game Anda ke Amazon GameLift.

    Versi OpenSSL yang Anda paketkan dengan build game Anda harus sesuai dengan versi yang digunakan mesin game saat membangun server game. Pastikan Anda mengemas versi OpenSSL yang benar dengan build server game Anda. Untuk OS Windows, format OpenSSL adalah. `.dll`

    > **Note**
    >
    > Package DLL OpenSSL di build server game Anda. Pastikan untuk mengemas versi OpenSSL yang sama dengan yang Anda gunakan saat membangun server game.
    >
    > - `libssl-1_1-x64.dll`
    >
    >   `libcrypto-1_1-x64.dll`

Package dependensi Anda bersama dengan server game Anda yang dapat dieksekusi di root file zip. Misalnya, `openssl-lib` dll harus berada di direktori yang sama dengan file. `.exe`

Langkah selanjutnya

Anda telah mengonfigurasi dan menyiapkan lingkungan Unreal Engine Anda, dan sekarang Anda dapat mulai mengintegrasikan Amazon GameLift ke dalam game Anda.

Untuk informasi selengkapnya tentang menambahkan Amazon GameLift ke game Anda, lihat berikut ini:

- Tambahkan Amazon GameLift ke server game Anda
- Referensi SDK GameLift server Amazon untuk Unreal Engine

Untuk petunjuk tentang menguji game Anda, lihat Uji integrasi Anda menggunakan GameLift Anywhere armada Amazon.

Integrasikan Amazon GameLift ke dalam proyek Unity

Topik ini menjelaskan cara menyiapkan plugin Amazon GameLift C# Server SDK untuk Unity dan mengintegrasikannya ke dalam proyek game Anda.

Sumber daya tambahan:

- Situs unduhan SDK GameLift server Amazon
- Referensi SDK 5.x GameLift server Amazon untuk C # dan Unity
- the section called “Dukungan pengembangan”

Prasyarat

Untuk menggunakan plugin SDK server Amazon GameLift C # untuk Unity, Anda memerlukan komponen berikut:

- Lingkungan pengembangan dan versi Unity Editor yang didukung plugin (lihatDukungan pengembangan dengan Amazon GameLift). Untuk informasi tentang versi Unity, lihat Persyaratan sistem untuk Unity dalam dokumentasi Unity.
- Plugin SDK GameLift server Amazon untuk paket Unity. Paket ini termasuk server SDK 5+ untuk C#. Anda dapat mengunduh paket dari situs ini: Memulai dengan Amazon GameLift.
- Registri cakupan pihak ketiga. UnityNuGet Alat ini mengelola DLL pihak ketiga. Untuk informasi selengkapnya, lihat repositori UnityNuGetGithub.

Mengatur UnityNuGet

Jika Anda belum UnityNuGet menyiapkan proyek game Anda, gunakan langkah-langkah berikut untuk menginstal alat menggunakan manajer paket Unity. Atau, Anda dapat menggunakan NuGet CLI untuk mengunduh DLL secara manual. Untuk informasi selengkapnya, lihat SDK server Amazon GameLift C# untuk Unity. `README`

Untuk mengintegrasikan UnityNuGet ke dalam proyek game Anda

1. Dengan proyek Anda terbuka di Unity Editor, buka menu utama dan pilih Edit, Pengaturan Proyek. Dari opsi, pilih bagian Package Manager dan buka grup Scoped Registries.
2. Pilih tombol + dan masukkan nilai berikut untuk UnityNuGet registri cakupan:

   ```
   Name: Unity NuGet
   URL: https://unitynuget-registry.azurewebsites.net
   Scope(s): org.nuget
   ```

3. Untuk pengguna versi Unity 2021:

   Setelah menyiapkan UnityNuGet, periksa `Assembly Version Validation` kesalahan yang ditampilkan di konsol Unity. Kesalahan ini terjadi jika pengalihan pengikatan untuk rakitan yang diberi nama kuat dalam NuGet paket tidak diselesaikan dengan benar ke jalur dalam proyek Unity Anda. Untuk mengatasi masalah ini, konfigurasikan validasi versi perakitan Unity:

   a. Di Editor Unity, buka menu utama dan pilih Edit, Pengaturan Proyek, dan buka bagian Player.

   b. Hapus pilihan opsi Validasi Versi Majelis.

Pasang plugin

Gunakan prosedur berikut untuk menginstal plugin SDK server Amazon GameLift C # untuk Unity dan mengkonfigurasi log4net logging.

Untuk memasang plugin

1. Dengan proyek Anda terbuka di Unity Editor, buka menu utama dan pilih Window, Package Manager.
2. Pilih tombol + untuk menambahkan paket baru. Pilih opsi Tambahkan paket dari tarball.
3. Di Pilih paket pada disk, cari plugin Amazon GameLift C# Server SDK untuk file unduhan Unity, dan pilih file Amazon GameLift Server SDK`.tgz`. Pilih Buka untuk menginstal plugin.

SDK GameLift server Amazon menggunakan kerangka log4net untuk mengeluarkan pesan log. Ini dikonfigurasi untuk menampilkan pesan ke terminal server build secara default, tetapi Unity memerlukan konfigurasi untuk menambahkan dukungan pencatatan file. Anda dapat menambahkan dukungan ini ke project Anda dengan mengimpor sampel yang disediakan di dalam paket Amazon GameLift Server SDK. Gunakan prosedur berikut untuk menambahkan sampel dan mengkonfigurasi log4net:

Untuk mengkonfigurasi log4net untuk output file

1. Dengan proyek Anda terbuka di Unity Editor, buka menu utama dan pilih Window, Package Manager.
2. Dari menu tarik-turun, pilih Packages: In Project, lalu pilih Amazon GameLift Server SDK dari daftar paket. Ini membuka detail paket.
3. Dalam detail paket, pilih opsi grup Sampel dan tekan Impor.
4. `log4net.config`File dan `LoggingConfiguration.cs` skrip yang menyertainya secara otomatis mengeksekusi konfigurasi, yang sekarang diatur di folder proyek. `Assets/Samples`

> **Note**
>
> Jika Anda perlu memindahkan `log4net.config` file Anda ke folder lain dalam proyek, maka Anda juga harus memperbarui filepath file konfigurasi dalam skrip `LoggingConfiguration.cs` dengan jalur baru. Untuk informasi selengkapnya, lihat manual log4net tentang mengonfigurasi log4net.

Untuk petunjuk lebih rinci dan panduan pengujian, lihat yang `README` terletak di download plugin.

Siapkan GameLift Anywhere armada Amazon untuk pengujian

Anda dapat mengatur workstation pengembangan sebagai armada GameLift Anywhere hosting Amazon untuk menguji integrasi Amazon GameLift secara berulang. Dengan pengaturan ini, Anda dapat memulai proses server game di workstation Anda, mengirim permintaan bergabung pemain atau perjodohan ke Amazon GameLift untuk memulai sesi permainan, dan menghubungkan klien ke sesi permainan baru. Dengan workstation Anda sendiri yang disiapkan sebagai server hosting, Anda dapat memantau semua aspek integrasi game Anda dengan Amazon GameLift.

Untuk petunjuk tentang pengaturan workstation Anda, lihat Uji integrasi Anda menggunakan GameLift Anywhere armada Amazon untuk menyelesaikan langkah-langkah berikut:

1. Buat lokasi khusus untuk workstation Anda.
2. Buat GameLift Anywhere armada Amazon 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 baruAnywhere. 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 GameLift layanan Amazon. Simpan otentikasi pada komputasi workstation Anda sehingga proses server game Anda yang sedang berjalan dapat mengaksesnya.

Tambahkan kode GameLift server Amazon ke project Unity Anda

Server game Anda berkomunikasi dengan GameLift layanan Amazon untuk menerima instruksi dan melaporkan status yang sedang berlangsung. Untuk mencapai hal ini, Anda menambahkan kode server game yang menggunakan SDK GameLift server Amazon.

Contoh kode yang disediakan menggambarkan elemen integrasi dasar yang diperlukan. Ini menggunakan a `MonoBehavior` untuk mengilustrasikan inisialisasi server game sederhana dengan Amazon. GameLift Contohnya mengasumsikan bahwa server game berjalan pada GameLift Anywhere armada Amazon untuk pengujian. Ini termasuk kode untuk:

- Inisialisasi klien GameLift API Amazon. Sampel menggunakan versi `InitSDK()` dengan parameter server untuk Anywhere armada dan komputasi Anda. Gunakan WebSocket URL, ID armada, nama komputasi (ID host), dan token otentikasi, seperti yang didefinisikan dalam topik sebelumnya. Siapkan GameLift Anywhere armada Amazon untuk pengujian
- Menerapkan fungsi callback untuk menanggapi permintaan dari GameLift layanan Amazon, termasuk`OnStartGameSession`,`OnProcessTerminate`, dan`onHealthCheck`.
- Panggil ProcessReady () dengan port yang ditunjuk untuk memberi tahu GameLift layanan Amazon saat proses siap untuk menyelenggarakan sesi permainan.

Kode yang disajikan dalam topik ini membangun komunikasi dengan GameLift layanan Amazon dan. Hal ini juga mengimplementasikan satu set fungsi callback yang menanggapi permintaan dari. 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, bacaReferensi SDK GameLift server Amazon untuk C #.

Contoh ini menunjukkan cara menambahkan semua elemen yang diperlukan, seperti yang dijelaskan dalam Tambahkan Amazon GameLift ke server game Anda. Ini termasuk:

Untuk informasi selengkapnya tentang menambahkan GameLift fungsionalitas Amazon, lihat topik berikut:

- Tambahkan Amazon GameLift ke server game Anda
- Referensi SDK GameLift server Amazon untuk C #

```
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 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 a GameLift Anywhere fleet.
        //They are not required for a GameLift managed EC2 fleet.
        ServerParameters serverParameters = new ServerParameters(
            webSocketUrl,
            processId,
            hostId,
            fleetId,
            authToken);

        //InitSDK establishes a local connection with an Amazon GameLift agent
        //to enable further communication.
        var initSDKOutcome = GameLiftServerAPI.InitSDK(serverParameters);
        if (initSDKOutcome.Success)
        {
            //Implement callback functions
            ProcessParameters processParameters = new ProcessParameters(
            //Implement OnStartGameSession callback
                (gameSession) => {
                    //GameLift 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) => {
                    //GameLift 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
                    //GameLift 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 GameLift it is shutting
down.
                    GameLiftServerAPI.ProcessEnding();
                },
                () => {
                    //Implement callback function OnHealthCheck
                    //GameLift 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 GameLift
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 GameLift where to find game session
log files.
                    //At the end of a game session, GameLift 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 GameLift 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 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);
        }
    }
}
```

Sumber daya tambahan

Gunakan sumber daya berikut untuk menguji server game Anda dan memperluas fungsionalitasnya:

- Siapkan mesin pengembangan Anda sebagai armada Amazon GameLift Anywhere dan gunakan untuk pengujian lokal. Lihat Menguji integrasi server kustom Anda.
- Bangun server game Anda dan unggah build ke Amazon GameLift. Lihat Mengunggah build server khusus ke Amazon GameLift.
- Menerapkan build server game Anda ke armada EC2 GameLift terkelola Amazon. Lihat Membuat GameLift armada Amazon baru.

Uji integrasi Anda menggunakan GameLift Anywhere armada Amazon

Anda dapat menggunakan GameLift Anywhere armada Amazon untuk membangun dan menguji integrasi game Anda secara berulang dengan Amazon. GameLift Siapkan perangkat keras Anda sendiri sebagai Anywhere armada dengan koneksi ke GameLift layanan Amazon, lalu instal dan jalankan server game Anda di dalamnya. Gunakan aplikasi pengujian untuk menjalankan skenario seperti memulai/menghentikan sesi permainan, melacak koneksi pemain, dan menangani isi ulang perjodohan. Dengan Anywhere armada, Anda dapat memperbarui build server game Anda sesuai kebutuhan dan memiliki visibilitas penuh ke dalam aktivitas hosting.

Anda dapat GameLift Anywhere armada Amazon dengan game yang terintegrasi dengan Amazon GameLift Server SDK versi 5 atau lebih tinggi.

Topik

- Perkembangan awal
- Iterasi di server game Anda

Perkembangan awal

Anda telah mengembangkan game Anda dan mengintegrasikannya dengan SDK GameLift server Amazon. Untuk menguji integrasi Anda, Anda dapat mengunggah setiap iterasi baru build server game Anda ke Amazon GameLift dan membuat armada. Atau, menggunakan Anywhere armada dengan laptop pengembangan Anda memberi Anda cara yang lebih efisien untuk melakukan pengembangan dan pengujian berulang.

Gunakan prosedur berikut untuk membuat Anywhere armada dan memulai sesi permainan di laptop Anda menggunakan GameLift konsol Amazon atau AWS Command Line Interface (AWS CLI).

Console

1. Buka GameLiftkonsol Amazon.
2. Di panel navigasi, di bawah Hosting, pilih Lokasi.
3. Pilih Buat lokasi.
4. Dalam Buat lokasi kotak dialog, lakukan hal berikut:
   a. Masukkan nama Lokasi. Ini memberi label lokasi sumber daya komputasi Anda yang GameLift digunakan Amazon untuk menjalankan game Anda dalam Anywhere armada. Nama lokasi khusus harus dimulai dengan custom-.
   b. Pilih Create (Buat).
5. Untuk membuat Anywhere armada, lakukan hal berikut:
   a. Di panel navigasi, di bawah Hosting, pilih Armada.
   b. Pada halaman Armada, pilih Buat armada.
   c. Pada langkah Pilih jenis komputasi, pilih Di mana saja, lalu pilih Berikutnya.
   d. Pada langkah Define fleet details, tentukan armada baru Anda. Untuk informasi selengkapnya, lihat Buat GameLift armada Amazon baru.
   e. Pada langkah Pilih lokasi, pilih lokasi khusus yang Anda buat.
   f. Lengkapi langkah-langkah pembuatan armada yang tersisa untuk membuat Anywhere armada Anda.
6. Daftarkan laptop Anda sebagai sumber daya komputasi di armada yang Anda buat. Gunakan `register-compute`perintah (atau operasi `RegisterCompute`API). Sertakan yang `fleet-id` dibuat pada langkah sebelumnya dan tambahkan `compute-name` dan laptop Anda`ip-address`.

```
aws gamelift register-compute \
    --compute-name DevLaptop \
    --fleet-id fleet-1234 \
    --ip-address 10.1.2.3 \
    --location custom-location-1
```

Contoh keluaran:

```
Compute {
    FleetId = fleet-1234,
```

```
    ComputeName = DevLaptop,
    Status = ACTIVE,
    IpAddress = 10.1.2.3,
    GameLiftServiceSdkEndpoint = wss://12345678.execute-api.amazonaws.com/,
    Location = custom-location-1
}
```

7. Mulai sesi debug server game Anda.

   a. Dapatkan token otorisasi untuk laptop Anda di armada yang Anda buat. Gunakan `get-compute-auth-token`perintah (atau operasi `GetComputeAuthToken`API).

   ```
   aws gamelift get-compute-auth-token \
       --fleet-id fleet-1234 \
       --compute-name DevLaptop
   ```

   Contoh keluaran:

   ```
   ComputeAuthToken {
     FleetId = fleet-1234,
     ComputeName = DevLaptop,
     AuthToken = abcdefg123,
     ExpirationTime = 1897492857.11
   }
   ```

   b. Jalankan instance debug server game Anda yang dapat dieksekusi. Untuk menjalankan instance debug, server game Anda harus meneleponInitSDK(). Setelah proses siap untuk menjadi tuan rumah sesi permainan, panggilan server gameProcessReady().

8. Buat sesi game untuk menguji integrasi pertama Anda dengan Amazon GameLiftAnywhere. Gunakan `create-game-session`perintah (atau operasi `CreateGameSession`API). Tentukan lokasi kustom armada.

   ```
   aws gamelift create-game-session \
       --fleet-id fleet-1234 \
       --name DebugSession \
       --maximum-player-session-count 2 \
       --location custom-location-1
   ```

   Contoh keluaran:

```
GameSession {
    FleetId = fleet-1234,
    GameSessionId = 1111-1111,
    Name = DebugSession,
    IpAddress = 10.1.2.3,
    Port = 1024,
    ...
}
```

Amazon GameLift mengirimkan `onStartGameSession()` pesan ke proses server terdaftar Anda. Pesan berisi `GameSession` objek dari langkah sebelumnya dengan properti game, data sesi game, data mak comblang, dan lainnya tentang sesi permainan.

9. Tambahkan logika ke server game Anda sehingga proses server Anda merespons `onStartGameSession()` pesan dengan`ActivateGameSession()`. Operasi mengirimkan pengakuan ke Amazon GameLift bahwa server Anda menerima dan menerima pesan sesi buat game. Untuk informasi lebih lanjut lihat,Referensi SDK GameLift server Amazon.

Server game Anda sekarang menjalankan sesi permainan untuk Anda uji dan gunakan untuk iterasi. Untuk mempelajari cara melakukan iterasi di server game Anda, lanjutkan ke bagian berikutnya.

AWS CLI

1. Buat lokasi khusus menggunakan `create-location`perintah (atau operasi `CreateLocation`API). Lokasi khusus memberi label pada lokasi perangkat keras Anda yang GameLift digunakan Amazon untuk menjalankan game Anda dalam Anywhere armada.

```
aws gamelift create-location \
    --location-name custom-location-1
```

Contoh keluaran:

```
{
    Location {
        LocationName = custom-location-1
    }
}
```

2. Buat Anywhere armada dengan lokasi khusus Anda menggunakan `create-fleet`perintah (atau operasi `CreateFleet`API). Amazon GameLift membuat armada di Wilayah asal Anda dan lokasi khusus yang Anda berikan.

```
aws gamelift create-fleet \
    --name LaptopFleet \
    --compute-type ANYWHERE \
    --locations "location=custom-location-1"
```

Contoh keluaran:

```
Fleet {
    Name = LaptopFleet,
    ComputeType = ANYWHERE,
    FleetId = fleet-1234,
    Status = ACTIVE
    ...
}
```

3. Daftarkan laptop Anda sebagai sumber daya komputasi di armada yang Anda buat. Gunakan `register-compute`perintah (atau operasi `RegisterCompute`API). Sertakan yang `fleet-id` dibuat pada langkah sebelumnya dan tambahkan `compute-name` dan laptop Anda publik`ip-address`.

```
aws gamelift register-compute \
    --compute-name DevLaptop \
    --fleet-id fleet-1234 \
    --ip-address 10.1.2.3 \
    --location custom-location-1
```

Contoh keluaran:

```
Compute {
    FleetId = fleet-1234,
    ComputeName = DevLaptop,
    Status = ACTIVE,
    IpAddress = 10.1.2.3,
    GameLiftServiceSdkEndpoint = wss://12345678.execute-api.amazonaws.com/,
    Location = custom-location-1
}
```

4. Mulai sesi debug server game Anda.

   a. Dapatkan token otorisasi untuk laptop Anda di armada yang Anda buat. Gunakan `get-compute-auth-token`perintah (atau operasi `GetComputeAuthToken`API).

   ```
   aws gamelift get-compute-auth-token \
       --fleet-id fleet-1234 \
       --compute-name DevLaptop
   ```

   Contoh keluaran:

   ```
   ComputeAuthToken {
     FleetId = fleet-1234,
     ComputeName = DevLaptop,
     AuthToken = abcdefg123,
     ExpirationTime = 1897492857.11
   }
   ```

   b. Jalankan instance debug server game Anda yang dapat dieksekusi. Untuk menjalankan instance debug, server game Anda harus meneleponInitSDK(). Setelah proses siap untuk menjadi tuan rumah sesi permainan, panggilan server gameProcessReady().

5. Buat sesi game untuk menguji integrasi pertama Anda dengan Amazon GameLiftAnywhere. Gunakan `create-game-session`perintah (atau operasi `CreateGameSession`API).

   ```
   aws gamelift create-game-session \
       --fleet-id fleet-1234 \
       --name DebugSession \
       --maximum-player-session-count 2
   ```

   Contoh keluaran:

   ```
   GameSession {
       FleetId = fleet-1234,
       GameSessionId = 1111-1111,
       Name = DebugSession,
       IpAddress = 10.1.2.3,
       Port = 1024,
       ...
   }
   ```

Amazon GameLift mengirimkan `onStartGameSession()` pesan ke proses server terdaftar Anda. Pesan berisi `GameSession` objek dari langkah sebelumnya dengan properti game, data sesi game, data mak comblang, dan lainnya tentang sesi permainan.

6. Tambahkan logika ke server game Anda sehingga proses server Anda merespons `onStartGameSession()` pesan dengan`ActivateGameSession()`. Operasi mengirimkan pengakuan ke Amazon GameLift bahwa server Anda menerima dan menerima pesan sesi buat game. Untuk informasi lebih lanjut lihat,Referensi SDK GameLift server Amazon.

Server game Anda sekarang menjalankan sesi permainan untuk Anda uji dan gunakan untuk iterasi. Untuk mempelajari cara melakukan iterasi di server game Anda, lanjutkan ke bagian berikutnya.

Iterasi di server game Anda

Dalam kasus penggunaan ini, pertimbangkan skenario di mana Anda telah menyiapkan dan menguji server game Anda dan menemukan bug. Dengan Amazon GameLiftAnywhere, Anda dapat melakukan iterasi pada kode Anda dan menghindari penyiapan berat menggunakan armada Amazon EC2.

1. Bersihkan yang ada`GameSession`, jika memungkinkan. Jika server game mogok atau tidak akan meneleponProcessEnding(), Amazon GameLift membersihkan server game `GameSession` setelah server game berhenti mengirim pemeriksaan kesehatan.
2. Buat perubahan kode ke server game Anda, kompilasi, dan persiapkan untuk pengujian berikutnya.
3. AnywhereArmada Anda sebelumnya masih aktif dan laptop Anda masih terdaftar sebagai sumber daya komputasi di armada. Untuk memulai pengujian lagi, buat instance debug baru.

    a. Ambil token otorisasi untuk laptop Anda di armada yang Anda buat. Gunakan `get-compute-auth-token`perintah (atau operasi `GetComputeAuthToken`API).

    ```
    aws gamelift get-compute-auth-token \
        --fleet-id fleet-1234 \
        --compute-name DevLaptop
    ```

    Contoh keluaran:

```
ComputeAuthToken {
  FleetId = fleet-1234,
  ComputeName = DevLaptop,
  AuthToken = hijklmnop456,
  ExpirationTime = 1897492857.11
}
```

b. Jalankan instance debug server game Anda yang dapat dieksekusi. Untuk menjalankan instance debug, server game Anda harus meneleponInitSDK(). Setelah proses siap untuk menjadi tuan rumah sesi permainan, panggilan server gameProcessReady().

4. Armada Anda sekarang memiliki proses server yang tersedia. Buat sesi permainan Anda dan lakukan tes berikutnya. Gunakan create-game-sessionperintah (atau operasi CreateGameSessionAPI).

```
aws gamelift create-game-session \
    --fleet-id fleet-1234 \
    --name SecondDebugSession \
    --maximum-player-session-count 2
```

Amazon GameLift mengirimkan onStartGameSession() pesan ke proses server terdaftar Anda. Pesan berisi GameSession objek dari langkah sebelumnya dengan properti game, data sesi game, data mak comblang, dan lainnya tentang sesi permainan.

5. Tambahkan logika ke server game Anda sehingga proses server Anda merespons onStartGameSession() pesan denganActivateGameSession(). Operasi mengirimkan pengakuan ke Amazon GameLift bahwa server Anda menerima dan menerima pesan sesi buat game. Untuk informasi lebih lanjut lihat,Referensi SDK GameLift server Amazon.

Setelah selesai menguji server game, Anda dapat terus menggunakan Amazon GameLift untuk manajemen armada dan server game Anda. Untuk informasi selengkapnya, lihat Membuat GameLift Anywhere armada Amazon.

Menguji integrasi Anda menggunakan Amazon GameLift Local

> **Note**
>
> Gunakan prosedur pengujian ini jika Anda menggunakan versi SDK GameLift server Amazon yang versi 4.x atau versi lebih lama. Paket SDK server Anda menyertakan versi Amazon

GameLift Local yang kompatibel. Jika Anda menggunakan SDK server versi 5.x, lihat Uji integrasi Anda menggunakan GameLift Anywhere armada Amazon pengujian lokal dengan armada Amazon GameLiftAnywhere.

Gunakan Amazon GameLift Local untuk menjalankan versi terbatas dari GameLift layanan Amazon yang dikelola di perangkat lokal dan menguji integrasi game Anda terhadapnya. Alat ini berguna ketika melakukan pengembangan berulang pada integrasi game Anda. Alternatif—mengunggah setiap build baru ke Amazon GameLift dan mengonfigurasi armada untuk meng-host game Anda—dapat memakan waktu 30 menit atau lebih setiap kali.

Dengan Amazon GameLift Local, Anda dapat memverifikasi hal berikut:

- Server game Anda terintegrasi dengan benar dengan Server SDK dan berkomunikasi dengan benar dengan GameLift layanan Amazon 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 dan dapat mengambil informasi tentang sesi game yang ada, memulai sesi permainan baru, bergabung dengan pemain ke game, dan terhubung ke sesi permainan.

Amazon GameLift Local adalah alat baris perintah yang memulai versi mandiri dari layanan Amazon yang dikelola. GameLift Amazon GameLift Local juga menyediakan log peristiwa berjalan inisialisasi proses server, pemeriksaan kesehatan, serta panggilan dan respons API. Amazon GameLift Local mengenali subset tindakan AWS SDK untuk Amazon. GameLift Anda dapat melakukan panggilan dari AWS CLI atau dari klien game Anda. Semua tindakan API bekerja secara lokal seperti yang dilakukan di layanan GameLift web Amazon.

Setiap proses server seharusnya hanya menjadi tuan rumah satu sesi permainan. Sesi permainan adalah executable yang Anda gunakan untuk terhubung ke Amazon GameLift Local. Ketika sesi permainan selesai, Anda harus menelepon `GameLiftServerSDK::ProcessEndning` dan kemudian keluar dari proses. Saat menguji secara lokal dengan Amazon GameLift Local, Anda dapat memulai beberapa proses server. Setiap proses akan terhubung ke Amazon GameLift Local. 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 lokal mendukung API berikut:

- CreateGameSession
- CreatePlayerSession
- CreatePlayerSessions
- DescribeGameSessions
- DescribePlayerSessions

Mengatur Amazon GameLift lokal

Amazon GameLift Local disediakan sebagai `.jar` file yang dapat dieksekusi yang dibundel dengan Server SDK. Hal ini dapat dijalankan pada Windows atau Linux dan digunakan dengan bahasa Amazon GameLift -didukung.

Sebelum menjalankan Local, Anda juga harus memiliki berikut yang diinstal.

- Pembuatan Amazon GameLift Server SDK versi 3.1.5 hingga 4.x.
- Java 8

Menguji server game

Jika Anda ingin menguji server game Anda saja, Anda dapat menggunakan AWS CLI untuk mensimulasikan panggilan klien game ke layanan Amazon GameLift Local. Ini memverifikasi bahwa server game Anda melakukan seperti yang diharapkan dengan berikut ini:

- Server game diluncurkan dengan benar dan menginisialisasi Amazon GameLift Server SDK.
- Sebagai bagian dari proses peluncuran, server game memberi tahu Amazon GameLift bahwa server siap untuk menyelenggarakan sesi game.
- Server game mengirimkan status kesehatan ke Amazon GameLift setiap menit saat berjalan.
- Server game menanggapi permintaan untuk memulai sesi game baru.

1. Mulai Amazon GameLift 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 server game GameLift terintegrasi Amazon Anda 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 layanan Amazon GameLift Local. Ini berarti server game Anda berhasil menginisialisasi Amazon GameLift Server SDK (dengan`InitSDK()`). Ini telah memanggil `ProcessReady()` dengan jalur log yang ditampilkan dan, jika berhasil, siap untuk meng-host sesi game. Saat server game berjalan, Amazon GameLift 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: "ProcessReadytidak 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. MulaiAWS 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 titik akhir layanan GameLift web Amazon. 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 AWS CLI Jendela prompt perintah, `AWS gamelift` menghasilkan tanggapan seperti yang didokumentasikan di Referensi Perintah AWS CLI.

4. Buat sesi permainan.

   DenganAWS 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 Local telah mengirim on`StartGameSession` callback 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 merespons dengan objek sesi game 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 berubah, gunakan AWS CLI untuk memanggil `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
    }
}
```

Menguji 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 Local. Anda dapat memverifikasi tindakan berikut:

- Klien game berhasil membuat permintaan AWS SDK ke layanan Amazon GameLift Local, termasuk untuk membuat sesi game, mengambil informasi tentang sesi game 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 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 server game GameLift terintegrasi Amazon Anda secara lokal. Lihat Menguji server game untuk detail lebih lanjut tentang log pesan.

3. Konfigurasikan klien game Anda untuk Lokal dan mulailah.

   Untuk menggunakan klien game Anda dengan layanan Amazon GameLift Local, Anda harus membuat perubahan berikut pada pengaturan klien game Anda, seperti yang dijelaskan dalamSiapkan Amazon GameLift di 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`.

- Atur kredensial AWS. 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 layanan AmazonGameLift.

4. Uji panggilan klien game ke GameLift layanan Amazon.

   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 menunjukkan saat Amazon GameLift Local meminta server game Anda untuk memulai sesi permainan (`onStartGameSession`callback) dan berhasil `ActivateGameSession` saat 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 memanggil `DescribePlayerSessions()`, status sesi pemain harus berubah dari Disimpan ke Aktif.

6. Verifikasi bahwa server game Anda melaporkan status game dan pemain ke GameLift layanan Amazon.

   GameLiftAgar Amazon dapat mengelola permintaan pemain dan melaporkan metrik dengan benar, server game Anda harus melaporkan berbagai status kembali ke Amazon. GameLift

Verifikasi bahwa Local mencatat peristiwa yang terkait dengan tindakan berikut. Anda mungkin juga ingin menggunakan AWS CLI untuk melacak perubahan status.

- Pemain terputus dari sesi game — Pesan log GameLift lokal Amazon harus menunjukkan bahwa panggilan `RemovePlayerSession()` server game Anda. 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 game berakhir — Pesan log GameLift lokal Amazon 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 Server SDK v4.0.1. Lihat Mengakhiri sesi game .

- Proses server dihentikan — Pesan log GameLift lokal Amazon harus menunjukkan bahwa panggilan `ProcessEnding()` server game Anda. Sebuah AWS CLI panggilan ke `DescribeGameSessions()` harus mencerminkan perubahan status dari `Active` ke `Terminated` (atau `Terminating`).

Variasi dengan lokal

Saat menggunakan Amazon GameLift Local, ingatlah hal-hal berikut:

- Berbeda dengan layanan GameLift web Amazon, Local tidak melacak status kesehatan server dan memulai on`ProcessTerminate` callback. Local hanya berhenti mencatat laporan kondisi untuk server game.
- Untuk panggilan ke SDK AWS, ID armada tidak divalidasi, dan dapat berupa nilai string apa pun yang memenuhi persyaratan parameter(`^fleet-\S+`).
- ID sesi game dibuat dengan Local 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
```

Mengintegrasikan game dengan Amazon GameLift Realtime Server

Topik ini memberikan ikhtisar solusi Amazon GameLift dengan Realtime Server yang dikelola. Ringkasan menjelaskan kapan solusi ini cocok untuk game Anda, dan bagaimana Realtime Server mendukung game multipemain.

Untuk peta jalan lengkap agar game Anda aktif dan berjalan, lihat. Peta jalan hosting yang GameLift dikelola Amazon

> **Tip**
>
> Untuk mencoba hosting server GameLift game Amazon, lihatMemulai dengan Amazon GameLift.

Apa itu server Realtime?

Server realtime adalah server ready-to-go game ringan yang GameLift disediakan Amazon untuk Anda gunakan dengan game multipemain Anda. Server waktu nyata menghapus proses pengembangan, pengujian, dan penyebaran server game khusus. Solusi ini dapat membantu meminimalkan waktu dan usaha yang diperlukan untuk menyelesaikan permainan Anda.

Fitur kunci

- Tumpukan jaringan penuh untuk interaksi klien dan server game
- Fungsionalitas server game inti
- Logika server yang dapat disesuaikan
- Pembaruan langsung untuk konfigurasi Realtime dan logika server
- FlexMatchperjodohan
- Kontrol sumber daya hosting yang fleksibel

Siapkan server Realtime dengan membuat armada dan menyediakan skrip konfigurasi. Untuk informasi selengkapnya tentang membuat server Realtime dan cara mempersiapkan klien game Anda, lihatSiapkan server Realtime.

Bagaimana Realtime Server mengelola sesi game

Anda dapat menambahkan logika khusus untuk manajemen sesi game dengan membangunnya ke dalam skrip Realtime. Anda dapat menulis kode untuk mengakses objek khusus server, menambahkan logika berdasarkan peristiwa menggunakan callback, atau menambahkan logika berdasarkan skenario non-peristiwa.

Cara klien dan server Realtime berinteraksi

Selama sesi permainan, klien game berinteraksi dengan mengirim pesan ke server Realtime melalui layanan backend. Layanan backend kemudian menyampaikan pesan di antara klien game untuk bertukar aktivitas, status permainan, dan data game yang relevan.

Selain itu, Anda dapat menyesuaikan cara client dan server berinteraksi dengan menambahkan logika game ke skrip Realtime. Dengan logika game khusus, server Realtime mungkin mengimplementasikan callback untuk memulai respons berdasarkan peristiwa.

Protokol komunikasi

Server realtime 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 mengirim pesan menggunakan TCP sebagai fallback.

Isi 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 pengembang. Klien game Anda bertindak berdasarkan OPCodes dalam pesan yang diterimanya.

Grup pemain

Realtime Servers menyediakan fungsionalitas untuk mengelola grup pemain. Secara default, Amazon GameLift 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.

Server Realtime dengan sertifikat TLS

Dengan Server Realtime, 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 server Realtime, server secara otomatis merespons dengan sertifikat TLS, yang divalidasi klien. Amazon GameLift menangani enkripsi menggunakan komunikasi TLS untuk TCP (WebSockets) dan DTLS untuk lalu lintas UDP.

Menyesuaikan server Realtime

Server Realtime berfungsi sebagai server relay stateless. Server Realtime menyampaikan paket pesan dan data game antara klien game yang terhubung ke game. Namun, server Realtime 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 skrip Realtime. Dengan logika permainan, misalnya, Anda dapat membangun permainan stateful dengan tampilan server-otoritatif dari status permainan.

Amazon GameLift mendefinisikan sekumpulan 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 mengirimkan pesan dari pemain tertentu atau ke pemain target, atau lakukan pemrosesan tambahan sebagai tanggapan.
- Beri tahu semua pemain saat pemain meninggalkan grup atau memutuskan sambungan dari server.
- Lihat konten objek sesi game atau objek pesan, dan gunakan data.

Menerapkan dan memperbarui Server Realtime

Keuntungan utama dari Realtime Server adalah kemampuan untuk memperbarui skrip Anda kapan saja. Saat Anda memperbarui skrip, Amazon GameLift mendistribusikan versi baru ke semua sumber

daya hosting dalam hitungan menit. Setelah Amazon GameLift menerapkan skrip baru, semua sesi permainan baru yang dibuat setelah titik itu akan menggunakan versi skrip baru. (Sesi game yang ada akan terus menggunakan versi aslinya.)

Mulai mengintegrasikan game Anda dengan Server Realtime:

- Mengintegrasikan klien game untuk Server Realtime
- Membuat skrip Realtime

Mengintegrasikan klien game untuk Server Realtime

Topik ini menjelaskan bagaimana mempersiapkan klien game Anda untuk dapat bergabung dan berpartisipasi dalam sesi game yang GameLift di-host Amazon.

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 di-host di server Realtime dan bertukar pesan.

Menemukan atau membuat sesi permainan dan sesi pemain

Siapkan klien game Anda untuk menemukan atau memulai sesi permainan, meminta FlexMatch perjodohan, dan cadangan ruang untuk pemain dalam permainan dengan membuat sesi pemain. Sebagai praktik terbaik, buat layanan backend dan gunakan untuk membuat permintaan langsung ke GameLift layanan Amazon saat dipicu oleh aksi klien game. Layanan backend kemudian menyampaikan respons yang relevan kembali ke klien game.

1. Tambahkan AWS SDK ke klien game Anda, inisialisasi GameLift klien Amazon, dan konfigurasikan untuk menggunakan sumber daya hosting di armada dan antrean Anda. AWSSDK tersedia dalam beberapa bahasa; lihat Amazon GameLift SDKUntuk layanan klien khusus.
2. Tambahkan GameLift fungsionalitas ke layanan backend Anda. Untuk petunjuk lebih rinci, lihat Tambahkan Amazon GameLift 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 kemampuan untuk menempatkan sesi permainan baru dengan cepat dan cerdas, serta menggunakan data latensi pemain untuk meminimalkan jeda permainan. GameLift Minimal, layanan backend Anda harus dapat meminta

sesi game 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 game dan objek sesi pemain sebagai respons terhadap permintaan ke GameLift layanan Amazon. 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.

Terhubung ke game di Server Realtime

Aktifkan klien game Anda untuk connect langsung dengan sesi game yang di-host di server Realtime dan bertukar pesan dengan server dan dengan pemain lain.

1. Dapatkan SDK Klien Realtime, 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 koneksi klien/server untuk 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 Referensi API klien Server Realtime (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 Referensi API klien Server Realtime (C #): Callback 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 basic dengan Klien Basic Realtime (C#). Seperti yang ditunjukkan, contoh menginisialisasi objek klien Realtime, menyiapkan event handler dan mengimplementasikan callback sisi klien, menghubungkan ke server Realtime, mengirim pesan, dan memutusnya.

```
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 GameLift Realtime client SDK
     *
     * 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 GameLift 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 GameLift
```

```
            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 skrip Realtime

Untuk menggunakan Server Realtime untuk game Anda, Anda perlu menyediakan skrip (dalam bentuk beberapa JavaScript kode) untuk mengonfigurasi dan menyesuaikan armada Server Realtime secara opsional. Topik ini mencakup langkah-langkah kunci dalam membuat skrip Realtime. Setelah skrip siap, unggah ke GameLift layanan Amazon dan gunakan untuk membuat armada (lihatUnggah skrip Server Realtime ke Amazon GameLift).

Untuk menyiapkan skrip untuk digunakan dengan Server Realtime, tambahkan fungsionalitas berikut ke skrip Realtime Anda.

Mengelola siklus hidup sesi permainan (wajib)

Minimal, skrip Realtime harus menyertakan fungsionalitas `Init()`, yang mempersiapkan Server Realtime untuk memulai sesi game. Juga sangat disarankan agar Anda juga menyediakan cara untuk mengakhiri sesi game, untuk memastikan bahwa sesi game baru dapat terus dimulai di armada Anda.

fungsionalitas callback `Init()`, ketika dipanggil, dilewatkan objek sesi Realtime, yang berisi antarmuka untuk server Realtime. Lihat Antarmuka Server Realtime untuk detail lebih lanjut tentang antarmuka ini.

Untuk mengakhiri sesi game dengan cakap, skrip juga harus memanggil fungsionalitas server Realtime`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.

Server Realtime dengan konfigurasi paling basic--inisialisasi dan penghentian proses server--pada dasarnya bertindak sebagai server relai stateless. Server Realtime meneruskan pesan dan data game antara klien game yang terhubung ke game, tetapi tidak mengambil tindakan independen untuk memproses data atau menjalankan 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 game ke skrip Realtime Anda secara opsional. Misalnya, Anda dapat menjalankan perintah berikut. Kode contoh skrip memberikan ilustrasi. Lihat Referensi skrip Amazon GameLift Realtime Server.

- Tambahkan logika berbasis peristiwa. Menerapkan fungsionalitas callback untuk menanggapi peristiwa client-server. Lihat Callback skrip untuk Server Realtime 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 sumber daya AWS. Untuk detailnya, lihat Berkomunikasi dengan sumber daya AWS lain dari armada.
- Izinkan logika game mengakses informasi armada untuk instans yang sedang dijalankan. Untuk detailnya, lihat Mendapatkan data armada untuk instans Amazon GameLift.

Contoh skrip Server Realtime

Contoh ini menggambarkan script basic yang diperlukan untuk men-deploy Server Realtime ditambah beberapa logika kustom. 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 to 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 GameLift
// 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
```

```
};
```

Mengintegrasikan game dengan GameLift plugin Amazon untuk Unity

Topik di bagian ini menjelaskan GameLift plugin Amazon untuk Unity dan cara menggunakannya untuk mempersiapkan proyek game multipemain Anda untuk hosting dengan Amazon GameLift. Bekerja sepenuhnya di lingkungan pengembangan Unity Anda dengan alur kerja terpandu plugin untuk melengkapi persyaratan dasar hosting dengan Amazon GameLift.

Amazon GameLift adalah layanan yang dikelola sepenuhnya yang memungkinkan pengembang game mengelola dan menskalakan server game khusus untuk game multipemain berbasis sesi. Untuk informasi selengkapnya tentang GameLift hosting Amazon, lihatCara GameLift kerja Amazon.

- GameLift Plugin Amazon untuk panduan Unity untuk server SDK 5.x, versi 2.0.0, bekerja dengan server SDK 5.x dan mendukung Amazon. GameLift Anywhere
- GameLift Plugin Amazon untuk panduan Unity untuk server SDK 4.x, versi 1.0.0, bekerja dengan server SDK 4.x atau yang lebih lama. Versi ini menggunakan Amazon GameLift Local untuk pengujian integrasi.

GameLift Plugin Amazon untuk panduan Unity untuk server SDK 5.x

Amazon GameLift menyediakan alat untuk mempersiapkan server game multipemain Anda untuk bekerja dengan Amazon GameLift. GameLift Plugin Amazon untuk Unity memudahkan Anda mengintegrasikan Amazon GameLift ke dalam proyek game Unity Anda, menguji integrasi Anda dengan Amazon GameLift Anywhere, dan menyebarkan GameLift sumber daya Amazon untuk hosting cloud.

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.

Topik

- Tentang plugin
- Alur kerja plugin
- Instal plugin untuk Unity

- Mengatur profil AWS pengguna
- Siapkan game Anda untuk pengujian lokal dengan Amazon GameLift Anywhere
- Terapkan game Anda ke cloud hosting dengan armada EC2 terkelola

Tentang plugin

Plugin untuk Unity memberikan pengalaman memulai yang efisien untuk mengintegrasikan dan menghosting game multipemain Unity Anda dengan Amazon. GameLift Anda dapat memanfaatkan fungsionalitas plugin dan komponen pra-bangun untuk membuat game Anda aktif dan berjalan dengan cepat.

Plugin menambahkan alat dan fungsionalitas ke editor Unity. Gunakan alur kerja yang dipandu untuk mengintegrasikan Amazon GameLift ke dalam proyek game Anda, mengujinya secara lokal, dan kemudian menyebarkan server game ke hosting cloud Amazon GameLift .

Gunakan solusi hosting bawaan plugin untuk menyebarkan game Anda. Siapkan armada Amazon GameLift Anywhere dengan workstation lokal Anda sebagai host. Untuk cloud hosting, pilih dari dua skenario penerapan umum yang menyeimbangkan latensi pemain, ketersediaan sesi game, dan biaya dengan cara yang berbeda. Satu skenario termasuk mak FlexMatch comblang sederhana dan set aturan. Gunakan skenario ini untuk menempatkan solusi hosting siap produksi dasar, lalu optimalkan dan sesuaikan sesuai kebutuhan.

Plugin ini mencakup komponen-komponen ini:

- Modul plugin untuk editor Unity. Ketika plugin diinstal, item menu utama baru memberi Anda akses ke GameLift fungsionalitas Amazon.
- Pustaka C# untuk API GameLift layanan Amazon dengan fungsionalitas sisi klien.
- Pustaka C# untuk SDK GameLift server Amazon (versi 5.x).
- Cicipi konten game, termasuk aset dan adegan, sehingga Anda dapat mencoba Amazon GameLift meskipun 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.

Alur kerja plugin

Langkah-langkah berikut menjelaskan pendekatan khas untuk mengintegrasikan dan menerapkan proyek game dengan GameLift plugin Amazon untuk Unity. Anda menyelesaikan langkah-langkah ini dengan bekerja di editor Unity dan kode game Anda.

1. Buat profil pengguna yang menautkan ke AWS akun Anda dan memberikan kredensyal akses untuk pengguna akun yang valid dengan izin untuk menggunakan Amazon. GameLift
2. Tambahkan kode server ke proyek game Anda untuk membangun komunikasi antara server game yang sedang berjalan dan GameLift layanan With Amazon.
3. Tambahkan kode klien ke proyek game Anda yang memungkinkan klien game mengirim permintaan ke Amazon GameLift untuk memulai atau bergabung dengan sesi game dan kemudian terhubung ke server game.
4. Gunakan alur kerja Anywhere untuk menyiapkan workstation lokal Anda sebagai host Anywhere untuk server game Anda. Luncurkan server dan klien game Anda secara lokal, sambungkan ke sesi game, dan uji integrasi Anda.
5. Gunakan alur kerja hosting EC2 untuk mengunggah server game terintegrasi Anda dan menerapkan solusi hosting cloud. Saat server game Anda siap, luncurkan klien game Anda secara lokal, sambungkan ke sesi game dan masuk, dan mainkan game.

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 mengenalAWS, tindakan mungkin tercakup dalam Tingkat AWS Gratis.

Instal plugin untuk Unity

Bagian ini menjelaskan cara menambahkan plugin ke proyek Unity. Setelah plugin diinstal, fungsionalitas plugin tersedia ketika Anda memiliki proyek terbuka di editor Unity.

Sebelum Anda mulai

Inilah yang Anda butuhkan untuk menggunakan GameLift plugin Amazon untuk Unity:

- Unity untuk Windows 2022 LTS atau Unity untuk macOS
- GameLift Plugin Amazon untuk unduhan Unity. [Situs unduhan] Unduhan mencakup dua paket:
  - Plugin GameLift mandiri Amazon untuk Unity
  - Amazon GameLift C# server SDK for Unity

- Microsoft Visual Studio 2019 atau yang lebih baru.
- Proyek game multipemain dengan kode game C #.
- Registri cakupan pihak ketiga. UnityNuGet Alat ini mengelola DLL pihak ketiga. Untuk informasi selengkapnya, lihat repositori UnityNuGetGithub.

Tambahkan plugin ke proyek game Anda

Selesaikan tugas-tugas berikut, bekerja di editor Unity dan file proyek game Anda.

Langkah 1: Tambahkan UnityNuGet ke proyek game Anda

Jika Anda belum UnityNuGet menyiapkan proyek game Anda, gunakan langkah-langkah berikut untuk menginstal alat menggunakan manajer paket Unity. Atau, Anda dapat menggunakan NuGet CLI untuk mengunduh DLL secara manual. Untuk informasi selengkapnya, lihat SDK server Amazon GameLift C# untuk Unity. `README`

1. Dengan proyek Anda terbuka di editor Unity, buka menu utama dan pilih Edit, Pengaturan Proyek. Dari opsi, pilih bagian Package Manager dan buka grup Scoped Registries.
2. Pilih tombol + dan masukkan nilai berikut untuk UnityNuGet registri cakupan:

```
Name: Unity NuGet
URL: https://unitynuget-registry.azurewebsites.net
Scope(s): org.nuget
```

Untuk pengguna versi Unity 2021:

Setelah menyiapkan UnityNuGet, periksa `Assembly Version Validation` kesalahan yang ditampilkan di konsol Unity. Kesalahan ini terjadi jika pengalihan pengikatan untuk rakitan yang diberi nama kuat dalam NuGet paket tidak diselesaikan dengan benar ke jalur dalam proyek Unity Anda. Untuk mengatasi masalah ini, konfigurasikan validasi versi perakitan Unity:

1. Di editor Unity, buka menu utama dan pilih Edit, Pengaturan Proyek, dan buka bagian Player.
2. Hapus pilihan opsi Validasi Versi Perakitan.

Langkah 2: Tambahkan plugin dan paket SDK server C

1. Buka zip GameLift plugin Amazon untuk Unity download, yang berisi kedua paket.

2. Dengan proyek Anda terbuka di Unity Editor, buka menu utama dan pilih Window, Package Manager.
3. Pilih tombol + untuk menambahkan paket baru. Pilih opsi Tambahkan paket dari tarball.
4. Di Pilih paket pada disk, cari plugin Amazon GameLift C# Server SDK untuk file unduhan Unity, dan pilih file. `com.amazonaws.gameliftserver.sdk-<version>.tgz` Pilih Buka untuk menginstal plugin.
5. Di Pilih paket pada disk, cari plugin GameLift mandiri Amazon untuk file unduhan Unity, dan pilih file`com.amazonaws.gamelift-<version>.tgz`. Pilih Buka untuk menginstal plugin.
6. Verifikasi bahwa plugin mandiri ditambahkan ke proyek Anda. Kembali ke jendela editor Unity. Periksa menu utama untuk tombol GameLift menu Amazon baru.

Langkah 3: Impor permainan sampel (opsional)

Plugin untuk Unity hadir dengan serangkaian aset game sampel, termasuk adegan, yang dapat Anda tambahkan ke proyek game Anda. Mengimpor game contoh memberi Anda jalur cepat untuk menguji, membangun, dan menerapkan game multipemain sederhana dengan Amazon. GameLift Game sampel sudah sepenuhnya terintegrasi dengan Amazon GameLift SDK, sehingga Anda dapat melewati tugas integrasi dan menyelesaikan tugas alur kerja yang tersisa.

Saat menggunakan game contoh, Anda dapat mengatur dan bergabung dengan armada Amazon GameLift Anywhere yang dihosting secara lokal hanya dalam beberapa menit. Anda dapat menyebarkan game ke Amazon GameLift dan bergabung dengan game langsung yang dihosting di cloud dalam waktu kurang dari satu jam.

Untuk mengimpor contoh game:

1. Dengan proyek game Anda terbuka di Unity Editor, buka GameLift menu Amazon dan pilih Contoh Game, Impor Contoh Game.
2. Setelah file diimpor, buka GameLift menu Amazon lagi dan pilih Contoh Game, Inisialisasi Pengaturan. Langkah ini mengonfigurasi proyek Anda untuk membangun klien dan server game.

Ketika instalasi selesai, Anda akan melihat dua adegan baru ditambahkan ke proyek game Anda. Anda juga akan melihat beberapa aset proyek tambahan, termasuk GameLiftClientSettingsaset.

Untuk detail selengkapnya tentang UI dan gameplay sampel, lihat contoh readme game.

Mengatur profil AWS pengguna

Setelah menginstal plugin, atur profil dan tautkan ke pengguna AWS akun yang valid. Anda dapat mempertahankan beberapa profil, tetapi Anda hanya dapat memiliki satu profil aktif pada satu waktu. Setiap kali Anda bekerja di plugin, pilih profil untuk digunakan.

Mempertahankan beberapa profil memberi Anda kemampuan untuk beralih di antara skenario hosting yang berbeda. Misalnya, Anda dapat mengatur profil dengan AWS kredensyal yang sama tetapi Wilayah yang berbedaAWS. Atau Anda dapat mengatur profil dengan AWS akun yang berbeda atau dengan pengguna/set izin yang berbeda.

> **Note**
>
> Jika Anda telah menginstal AWS CLI di workstation Anda dan memiliki profil yang sudah dikonfigurasi, GameLift plugin Amazon dapat mendeteksinya dan akan 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.

Untuk mengatur AWS profil Anda

1. Di menu utama editor Unity, pilih Amazon GameLift dan pilih Set AWS Account Profiles. Tindakan ini membuka jendela plugin. Buka halaman Profil AWS Pengguna.
2. Jika plugin mendeteksi profil yang ada, Anda tidak akan diminta untuk membuatnya. Pilih profil yang ada atau pilih Tambahkan profil lain untuk membuat yang baru.
3. Jika plugin tidak mendeteksi profil yang ada, itu meminta Anda untuk membuatnya. Anda dapat membuat profil baru menggunakan AWS akun baru atau yang sudah ada.

   > **Note**
   >
   > Anda perlu menggunakan Konsol AWS Manajemen untuk membuat AWS akun baru dan membuat atau memperbarui pengguna dengan set izin yang tepat.

   Saat mengatur profil, Anda memerlukan informasi berikut:

- Akun AWS Jika Anda perlu membuat AWS akun baru, ikuti petunjuk untuk membuat akun. Lihat Membuat AWS akun untuk detail selengkapnya.
- Pengguna AWS akun dengan izin untuk menggunakan Amazon GameLift dan AWS layanan lain yang diperlukan. Lihat Siapkan Akun AWS petunjuk tentang cara menyiapkan pengguna AWS Identity and Access Management (IAM) dengan GameLift izin Amazon.
- Kredensyal untuk pengguna AndaAWS. Pengguna ini juga membutuhkan akses terprogram dengan kredensyal jangka panjang. Kredensyal ini terdiri dari ID kunci AWS akses dan kunci AWS rahasia. Lihat Dapatkan kunci akses Anda untuk detail selengkapnya.
- AWSWilayah. Ini adalah lokasi geografis tempat Anda ingin membuat AWS sumber daya untuk hosting. Selama pengembangan, sebaiknya gunakan wilayah yang dekat dengan lokasi fisik Anda untuk meminimalkan latensi. Lihat daftar AWSwilayah yang didukung.

4. Saat Anda memilih atau membuat profil, periksa status bootstrap profil dan ambil tindakan sesuai kebutuhan. Semua profil harus di-bootstrap untuk menggunakan fungsionalitas plugin Amazon GameLift .

Untuk bootstrap profil Anda:

Bootstrapping menunjuk bucket Amazon S3 untuk digunakan dengan profil pengguna yang dipilih. Amazon S3 adalah AWS layanan inti untuk penyimpanan data dan objek. Bucket digunakan untuk menyimpan konfigurasi proyek, membangun artefak, dan dependensi lainnya. Bucket tidak dibagi antara profil lain.

> **Note**
>
> Bootstrapping menciptakan AWS sumber daya baru dan mungkin menimbulkan biaya.

1. Saat melihat profil Anda di jendela plugin Profil AWS Pengguna, pilih profil yang ingin Anda gunakan. Pesan peringatan ditampilkan jika profil belum di-bootstrap.
2. Di bagian Bootstrap profil Anda, pilih profil dari daftar dropdown dan periksa status bootstrap. Jika status menunjukkan bahwa tidak ada bucket, pilih tombolnya Profil Bootstrap. Anda dapat menyetel nama bucket ke nama bucket baru, memasukkan bucket yang sudah ada yang dapat diakses, atau menyimpan nama yang dibuat secara otomatis.
3. Tunggu status bootstrap berubah menjadi “Aktif”. Hal ini dapat menghabiskan waktu beberapa menit. Ketika statusnya “Aktif”, Anda dapat menggunakan profil untuk bekerja dengan fitur plugin

Siapkan game Anda untuk pengujian lokal dengan Amazon GameLift Anywhere

Dalam alur kerja ini, Anda menambahkan kode game klien dan server untuk GameLift fungsionalitas Amazon 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 Anywhere:

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

Mengatur 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 GameLift menu Amazon 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

> **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 GameLift layanan Amazon 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 GameLiftServer file.cs. dan salin ke folder Server 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 C # (versi 5):

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

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

- Tambahkan Amazon GameLift ke server game Anda
- Referensi SDK 5.x GameLift server Amazon untuk C # dan Unity

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 GameLift layanan Amazon (di 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 GameLift layanan Amazon. Saat mengembangkan game untuk cloud hosting —atau jika 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

Untuk mengintegrasikan Amazon GameLift 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 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 kredensyal di mesin lokal Anda yang memungkinkan akses ke AWS SDK untuk. GameLift Klien game menggunakan kredensyal ini untuk mengautentikasi permintaan ke layanan Amazon. GameLift

> **Note**
>
> Profil kredensyal 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 memverifikasi bahwa status armada adalah iklan aktif 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 Unreal.

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 memverifikasi bahwa ia 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. Konfigurasikan 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 GameLift layanan Amazon.

    a. Jika Anda tidak mengimpor dan menginisialisasi game sampel, buat `GameLiftClientSettings` aset baru. Di menu utama editor Unity, pilih Assets, Create, GameLift, Client Settings. 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 bergabung 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-tokenperintah AWS CLI dengan ID AWS armada dan Wilayah Anywhere Anda. ID armada ditampilkan di Connect to Anywhere Fleet saat Anda membuat armada. AWSWilayah 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]
   ```

Terapkan game Anda ke cloud hosting dengan armada EC2 terkelola

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

Untuk memulai alur kerja Amazon EC2 yang GameLift dikelola Amazon:

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

Mengatur 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 GameLift menu Amazon 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

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

- Server game harus dapat berkomunikasi dengan GameLift layanan Amazon 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 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 C # (versi 5):

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

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

- Tambahkan Amazon GameLift ke server game Anda
- Referensi SDK 5.x GameLift server Amazon untuk C # dan Unity

Integrasikan kode klien Anda

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

Dalam alur kerja plugin untuk hosting pada armada EC2 terkelola, setiap skenario penerapan mencakup layanan backend pra-bangun yang mencakup komponen-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 ID pemain unik 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 GameLiftCoreApikelas 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 ke klien game Unity 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:
  - AWSarmada (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 API.
  - WebACL 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.
  - AWSantrian 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 API.
  - WebACL 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 armada EC2 terkelola Anda. Amazon GameLift menggunakan nama ini (bersama dengan ID armada) saat mencantumkan sumber daya di AWS konsol.
- Nama build: Berikan nama yang berarti untuk build server Anda. AWSmenggunakan nama ini untuk merujuk ke salinan build server Anda yang diunggah ke Amazon GameLift dan digunakan untuk penerapan.
- Parameter peluncuran: Masukkan instruksi opsional untuk dijalankan saat meluncurkan server yang dapat dieksekusi pada instance armada EC2 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 selama 40 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 instans AWS EC2. 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. Konfigurasikan 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 GameLift layanan Amazon.

a. Jika Anda tidak mengimpor dan menginisialisasi game sampel, buat `GameLiftClientSettings` aset baru. Di menu utama editor Unity, pilih Assets, Create, GameLift, Client Settings. 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 EC2 Terkelola. Tindakan ini memperbarui pengaturan klien game Anda untuk menggunakan armada EC2 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.

GameLift Plugin Amazon untuk panduan Unity untuk server SDK 4.x

> **Note**
>
> Topik ini memberikan informasi untuk versi sebelumnya dari GameLift plugin Amazon untuk Unity. Versi 1.0.0 (dirilis pada tahun 2021) menggunakan GameLift server Amazon SDK 4.x

> atau yang lebih lama. Untuk dokumentasi tentang versi terbaru plugin, yang menggunakan server SDK 5.x dan mendukung Amazon GameLift Anywhere, lihat. GameLift Plugin Amazon untuk panduan Unity untuk server SDK 5.x

Amazon GameLift menyediakan alat untuk mempersiapkan server game multipemain Anda agar berjalan di Amazon GameLift. GameLift Plugin Amazon untuk Unity membuatnya lebih mudah untuk mengintegrasikan Amazon GameLift ke dalam proyek game Unity Anda dan menyebarkan GameLift sumber daya Amazon untuk hosting cloud. Gunakan plugin Unity untuk mengakses Amazon GameLift API dan menerapkan AWS CloudFormation template untuk skenario game umum.

Setelah menyiapkan plugin, Anda dapat mencoba sampel Amazon GameLift Unity GitHub.

Topik

Integrasikan Amazon GameLift dengan proyek server game Unity

> **Note**
>
> Topik ini memberikan informasi untuk versi sebelumnya dari GameLift plugin Amazon untuk Unity. Versi 1.0.0 (dirilis pada tahun 2021) menggunakan GameLift server Amazon SDK 4.x atau yang lebih lama. Untuk dokumentasi tentang versi terbaru plugin, yang menggunakan server SDK 5.x dan mendukung Amazon GameLift Anywhere, lihat. GameLift Plugin Amazon untuk panduan Unity untuk server SDK 5.x

Topik ini membantu Anda mempersiapkan server game khusus untuk hosting di Amazon GameLift. Server game harus dapat memberi tahu Amazon GameLift tentang statusnya, untuk memulai

dan menghentikan sesi permainan saat diminta, dan untuk melakukan tugas lain. Untuk informasi selengkapnya, lihat Tambahkan Amazon GameLift ke server game Anda.

Prasyarat

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

- Menyiapkan peran layanan IAM untuk Amazon GameLift
- Instal plugin untuk Unity

Siapkan proses server baru

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

Siapkan komunikasi dengan Amazon GameLift dan laporkan bahwa proses server siap untuk menyelenggarakan sesi permainan.

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

Contoh kode berikut menunjukkan cara mengatur proses server sederhana dengan Amazon GameLift.

```
//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 GameLift plugin Amazon untuk Unity versi 1.0.0, yang menggunakan server SDK 4.x atau yang lebih lama.

Setelah inisialisasi game selesai, Anda dapat memulai sesi permainan.

1. Menerapkan fungsionalitas panggilan balik `onStartGameSession`. Amazon GameLift 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, lihatReferensi SDK GameLift server Amazon (C #): Tindakan.

Contoh kode berikut menggambarkan cara memulai sesi permainan dengan Amazon GameLift.

```
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 GameLift plugin Amazon untuk Unity versi 1.0.0, yang menggunakan server SDK 4.x atau yang lebih lama.

Beri tahu Amazon GameLift saat sesi game 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 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

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

Setelah Anda mengintegrasikan server game Anda dengan Amazon GameLift, unggah file build ke armada sehingga Amazon GameLift dapat menerapkannya untuk hosting game. Untuk informasi selengkapnya tentang cara mengunggah server Anda ke Amazon GameLift, lihatUnggah build server khusus ke Amazon GameLift.

Integrasikan Amazon GameLift dengan proyek klien game Unity

> **Note**
>
> Topik ini memberikan informasi untuk versi sebelumnya dari GameLift plugin Amazon untuk Unity. Versi 1.0.0 (dirilis pada tahun 2021) menggunakan GameLift server Amazon SDK 4.x atau yang lebih lama. Untuk dokumentasi tentang versi terbaru plugin, yang menggunakan server SDK 5.x dan mendukung Amazon GameLift Anywhere, lihat. GameLift Plugin Amazon untuk panduan Unity untuk server SDK 5.x

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

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

Server backend menangani tugas klien game berikut:

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

Topik

Prasyarat

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

- Siapkan Akun AWS
- Instal plugin untuk Unity
- Integrasikan Amazon GameLift dengan proyek server game Unity
- Menyiapkan GameLift armada Amazon

Inisialisasi klien game

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

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

1. Inisialisasi`AmazonGameLiftClient`. Panggil `AmazonGameLiftClient` dengan konfigurasi klien default atau konfigurasi khusus. Untuk informasi selengkapnya tentang cara mengkonfigurasi klien, lihatSiapkan Amazon GameLift di layanan backend.
2. Hasilkan id pemain unik untuk setiap pemain untuk terhubung ke sesi permainan. Untuk informasi selengkapnya, lihat Hasilkan ID pemain.

   Contoh berikut menunjukkan cara mengatur GameLift klien Amazon.

```
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 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 GameLift Client with default client configuration.
        aglc = new AmazonGameLiftClient(credentials, config);

    }
}
```

Buat sesi permainan pada armada tertentu

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

Tambahkan kode untuk memulai sesi game baru di armada yang Anda gunakan dan sediakan untuk pemain. Setelah GameLift Amazon membuat sesi permainan baru dan mengembalikan a`GameSession`, 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 GameLift plugin Amazon untuk Unity versi 1.0.0, yang menggunakan server SDK 4.x atau yang lebih lama.

Setelah GameLift Amazon 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 playerIdx, string playerSessionId)
{
    //Call ConnectPlayer with player ID and player session ID.
    return server.ConnectPlayer(playerIdx, playerSessionId);
}
```

Menghapus pemain dari sesi permainan

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

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

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

Contoh berikut menggambarkan cara menghapus sesi pemain.

```
public void DisconnectPlayer(int playerIdx)
{
```

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

Instal dan atur plugin

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

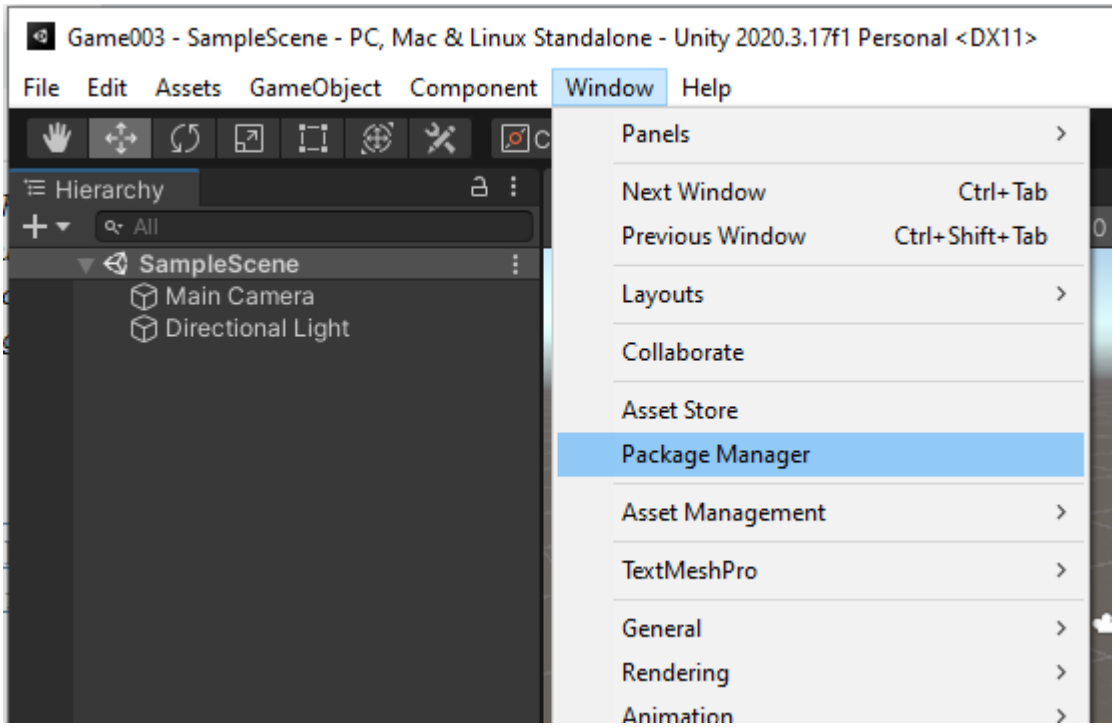
Bagian ini menjelaskan cara mengunduh, menginstal, dan mengatur GameLift plugin Amazon untuk Unity, versi 1.0.0.

Prasyarat

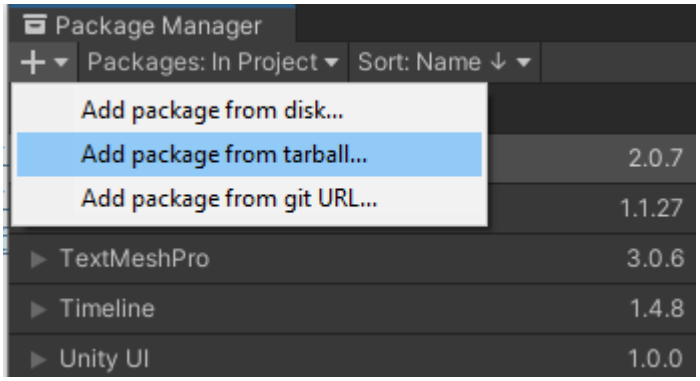
- Unity untuk Windows 2019.4 LTS, Windows 2020.3 LTS, atau Unity untuk macOS
- Versi Java saat ini
- Versi terkini dari.NET 4.x

Untuk mengunduh dan menginstal plugin untuk Unity

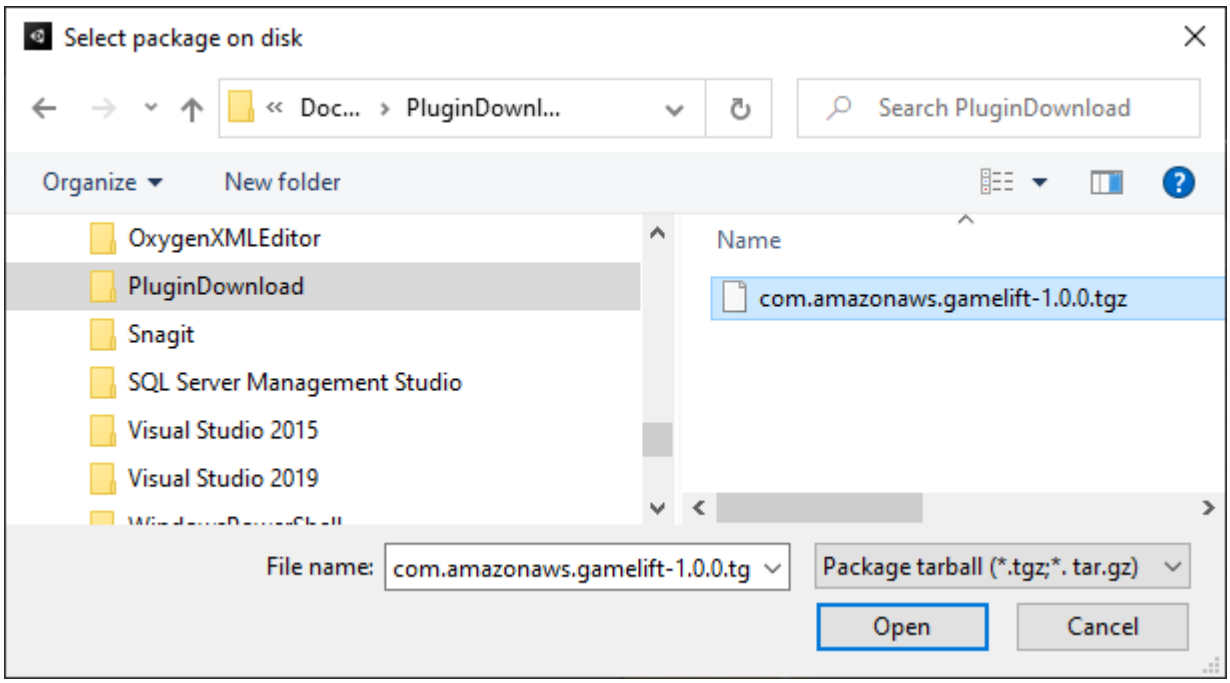
1. Unduh GameLift plugin Amazon untuk Unity. Anda dapat menemukan versi terbaru di GameLift plugin Amazon untuk halaman repositori Unity. Di bawah rilis terbaru, pilih Aset, lalu unduh `com.amazonaws.gamelift-`*`version`*`.tgz` file.
2. Luncurkan Unity dan pilih proyek.
3. Di bilah navigasi atas, di 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 file`com.amazonaws.gamelift-`*`version`*`.tgz` , lalu pilih Buka:



6. Setelah Unity memuat plug-in, Amazon GameLift muncul sebagai item baru di menu Unity. Mungkin perlu beberapa menit untuk menginstal dan mengkompilasi ulang skrip. Tab Pengaturan GameLift Plugin Amazon 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 GameLift plugin Amazon untuk Unity versi 1.0.0, yang menggunakan server SDK 4.x atau yang lebih lama.

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

Konfigurasikan pengujian lokal

1. Di jendela plugin untuk Unity, pilih tab Uji.
2. Di panel Uji, pilih Unduh Amazon GameLift 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 GameLift Plugin Amazon, pilih Jalur GameLift Lokal Amazon, 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.

Menerapkan skenario

> **Note**
>
> Topik ini mengacu pada GameLift plugin Amazon 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 GameLift disediakan Amazon dan cara menggunakannya.

Prasyarat

Untuk menerapkan skenario, Anda memerlukan peran IAM untuk layanan Amazon GameLift . Untuk informasi tentang cara membuat peran untuk Amazon GameLift, lihatSiapkan Akun AWS.

Setiap skenario memerlukan izin untuk sumber daya berikut:

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

Skenario

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

GameLift Plug-in Amazon 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 LambdaHandler 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 armada Amazon GameLift . Ini menciptakan sumber daya berikut:

- Kumpulan pengguna Amazon Cognito bagi pemain untuk mengautentikasi dan memulai permainan.
- AWS LambdaPawang 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 GameLift antrian Amazon dan mak comblang khusus untuk mengelompokkan pemain tertua di kolam tunggu. Ini menciptakan sumber daya berikut:

- Topik Layanan Pemberitahuan Sederhana Amazon tempat Amazon GameLift menerbitkan pesan. Untuk informasi selengkapnya tentang topik dan notifikasi SNS, lihatAtur 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 GameLift antrian Amazon dan mak comblang khusus dan mengonfigurasi 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, lihatPanduan desain GameLift armada Amazon.
- GameLift Antrian Amazon untuk menjaga ketersediaan server tetap tinggi dan biaya rendah. Untuk informasi selengkapnya dan praktik terbaik tentang antrian, lihat. Desain antrean sesi game

FlexMatch

Skenario ini menggunakan FlexMatch, layanan perjodohan terkelola, untuk mencocokkan pemain game bersama. Untuk informasi selengkapnya FlexMatch, lihat Apa itu Amazon GameLift FlexMatch. Skenario ini menciptakan sumber daya berikut:

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

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

Perbarui AWS kredensialnya

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

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

Untuk informasi selengkapnya tentang mengonfigurasi kredensil, lihat Memahami dan mendapatkan kredensil 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 kredensil baru atau memilih AWS kredensil yang ada.
    - Untuk membuat kredensil, 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 GameLift plugin Amazon 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 GameLift plugin Amazon untuk Unity versi 1.0.0, yang menggunakan server SDK 4.x atau yang lebih lama.

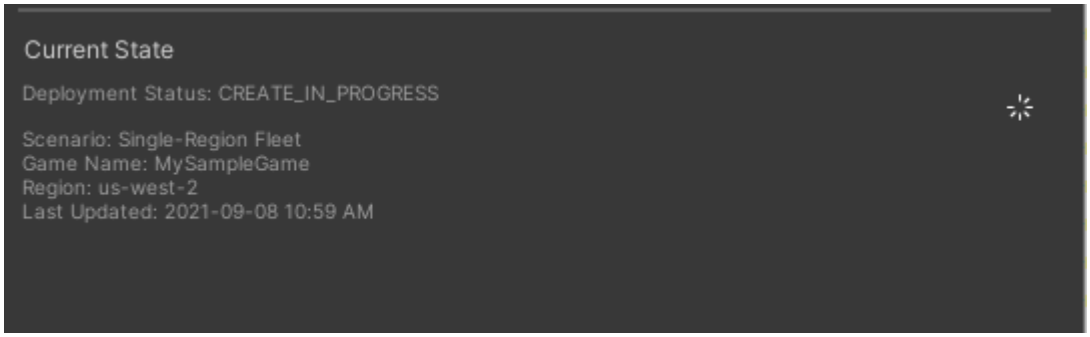
Anda dapat menggunakan skenario untuk menguji game Anda dengan Amazon GameLift. Setiap skenario menggunakan AWS CloudFormation template untuk membuat tumpukan dengan sumber daya yang diperlukan. Sebagian besar skenario memerlukan server game yang dapat dieksekusi dan membangun jalur. Saat Anda menerapkan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

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

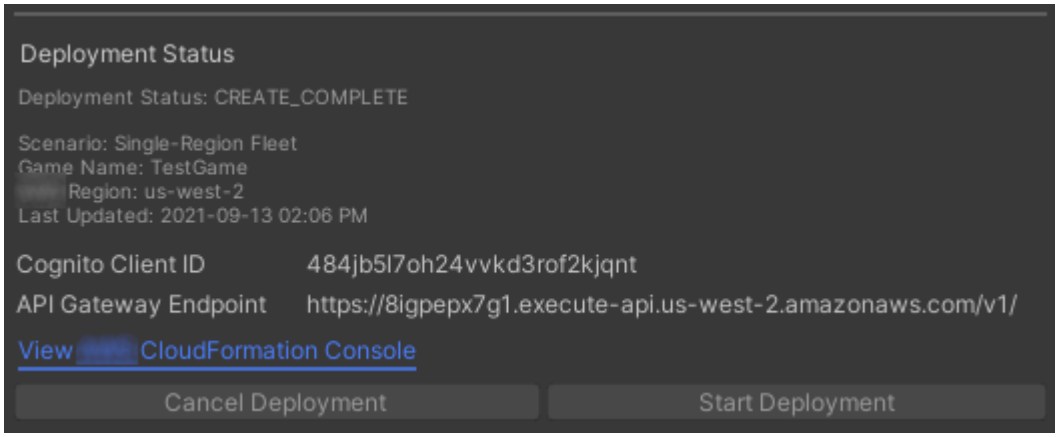
Untuk menyebarkan skenario

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 hingga 30 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 Coginito. 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 GameLift plugin Amazon 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 Plugin for 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`.

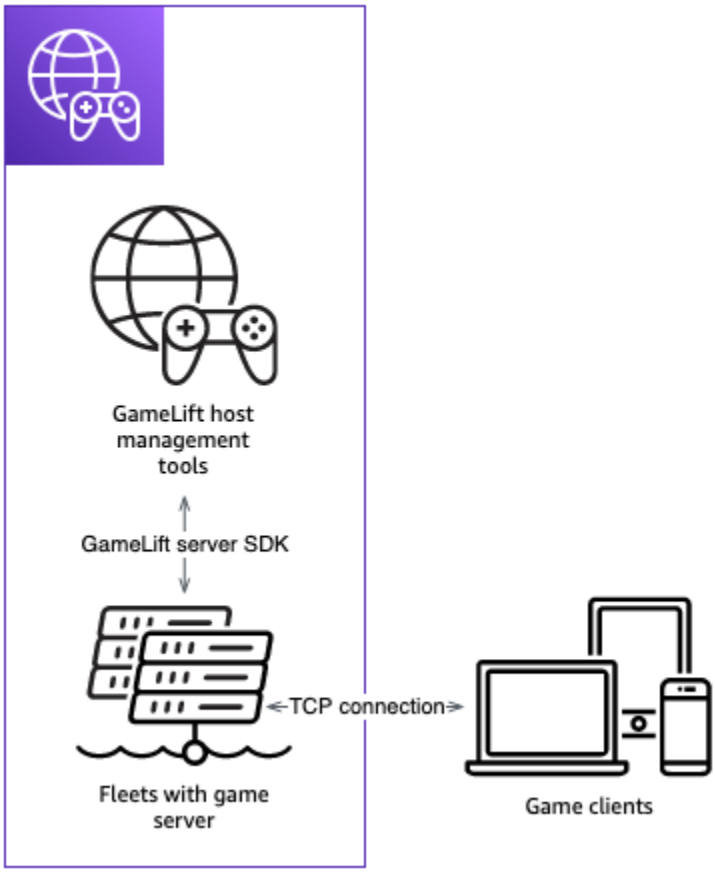
Integrasikan game dengan Amazon GameLift di Unity

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

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

- Integrasikan Amazon GameLift dengan proyek server game Unity
- Integrasikan Amazon GameLift dengan proyek klien game Unity

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



Impor dan jalankan contoh game

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

GameLift Plugin Amazon untuk Unity menyertakan contoh permainan yang dapat Anda gunakan untuk menjelajahi dasar-dasar mengintegrasikan game Anda dengan Amazon GameLift. Di bagian ini, Anda membangun klien game dan server game dan kemudian menguji secara lokal menggunakan Amazon GameLift Local.

Prasyarat

- Siapkan Akun AWS
- Instal dan atur plugin

Membangun dan menjalankan server game sampel

> **Note**
>
> Topik ini mengacu pada GameLift plugin Amazon 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, 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, lalu pilih Terapkan Pengaturan Pembuatan Server Sampel Windows atau Terapkan Pengaturan Pembuatan Server Sampel macOS. Setelah Anda mengonfigurasi pengaturan server game, Unity mengkompilasi 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 GameLift plugin Amazon 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, lalu pilih Apply Windows Sample Client Build Settings atau Apply macOS Sample Client Build Settings. Setelah pengaturan klien game dikonfigurasi, Unity akan mengkompilasi 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 GameLift Klien Amazon, 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 dengan Amazon GameLift.

Uji contoh permainan secara lokal

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

Jalankan contoh game yang Anda impor menggunakan Amazon GameLift Local.

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 Local. Jendela berisi pesan log termasuk pesan yang dikirim antara server game dan Amazon GameLift Local.
5. Luncurkan klien game. Temukan lokasi pembuatan dengan klien game sampel dan pilih file yang dapat dieksekusi.
6. Di Amazon GameLift Sample Game, berikan email dan kata sandi lalu pilih Masuk. Email dan kata sandi tidak divalidasi atau digunakan.
7. Di Amazon GameLift Sample Game, 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: playerIdx
 0 joined
2021-09-15T19:58:0335 PID:20728 Log SERVER: Frame: 8696 Connection accepted: playerIdx
 1 joined
2021-09-15T19:58:0338 PID:20728 Log SERVER: Frame: 8697 Msg rcvd from playerIdx 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 playerIdx 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 GameLift Lokal Amazon

```
12:55:26,000  INFO || - [SocketIOServer] main - Session store / pubsub factory used:
 MemoryStoreFactory (local session store only)
12:55:28,092  WARN || - [ServerBootstrap] main - Unknown channel option 'SO_LINGER' for
 channel '[id: 0xe23d0a14]'
12:55:28,101  INFO || - [SocketIOServer] nioEventLoopGroup-2-1 - SocketIO server
 started at port: 5757
12:55:28,101  INFO || - [SDKConnection] main - GameLift SDK server (communicates with
 your game server) has started on http://localhost:5757
```

```
12:55:28,120  INFO || - [SdkWebSocketServer] WebSocketSelector-20 - WebSocket Server
 started on address localhost/127.0.0.1:5759
12:55:28,166  INFO || - [StandAloneServer] main - GameLift Client server (listens for
 GameLift client APIs) has started on http://localhost:8080
12:55:28,179  INFO || - [StandAloneServer] main - GameLift server sdk http listener has
 started on http://localhost:5758
12:55:35,453  INFO || - [SdkWebSocketServer] WebSocketWorker-12 - onOpen
 socket: /?pID=20728&sdkVersion=4.0.2&sdkLanguage=CSharp and handshake /?
pID=20728&sdkVersion=4.0.2&sdkLanguage=CSharp
12:55:35,551  INFO || - [HostProcessManager] WebSocketWorker-12 - client connected with
 pID 20728
12:55:35,718  INFO || - [GameLiftSdkHttpHandler] GameLiftSdkHttpHandler-thread-0 -
 GameLift API to use: ProcessReady for pId 20728
12:55:35,718  INFO || - [ProcessReadyHandler] GameLiftSdkHttpHandler-thread-0 -
 Received API call for processReady from 20728
12:55:35,738  INFO || - [ProcessReadyHandler] GameLiftSdkHttpHandler-thread-0 -
 onProcessReady data: port: 33430
 12:55:35,739  INFO || - [HostProcessManager] GameLiftSdkHttpHandler-thread-0 -
 Registered new process with pId 20728
12:55:35,789  INFO || - [GameLiftSdkHttpHandler] GameLiftSdkHttpHandler-thread-0 -
 GameLift API to use: ReportHealth for pId 20728
12:55:35,790  INFO || - [ReportHealthHandler] GameLiftSdkHttpHandler-thread-0 -
 Received API call for ReportHealth from 20728
12:55:35,794  INFO || - [ReportHealthHandler] GameLiftSdkHttpHandler-thread-0 -
 ReportHealth data: healthStatus: true
 12:56:24,098  INFO || - [GameLiftHttpHandler] Thread-12 - API to use:
 GameLift.DescribeGameSessions
12:56:24,119  INFO || - [DescribeGameSessionsDispatcher] Thread-12 - Received API call
 to describe game sessions with input: {"FleetId":"fleet-123"}
12:56:24,241  INFO || - [GameLiftHttpHandler] Thread-12 - API to use:
 GameLift.CreateGameSession
12:56:24,242  INFO || - [CreateGameSessionDispatcher] Thread-12 - Received API call to
 create game session with input: {"FleetId":"fleet-123","MaximumPlayerSessionCount":4}
12:56:24,265  INFO || - [HostProcessManager] Thread-12 - Reserved process:
 20728 for gameSession: arn:aws:gamelift:local::gamesession/fleet-123/
gsess-59f6cc44-4361-42f5-95b5-fdb5825c0f3d
12:56:24,266  INFO || - [WebSocketInvoker] Thread-12 - StartGameSessionRequest:
 gameSessionId=arn:aws:gamelift:local::gamesession/fleet-123/
gsess-59f6cc44-4361-42f5-95b5-fdb5825c0f3d, fleetId=fleet-123, gameSessionName=null,
 maxPlayers=4, properties=[], ipAddress=127.0.0.1, port=33430, gameSessionData?=false,
 matchmakerData?=false, dnsName=localhost
12:56:24,564  INFO || - [CreateGameSessionDispatcher] Thread-12 - GameSession with
 id: arn:aws:gamelift:local::gamesession/fleet-123/gsess-59f6cc44-4361-42f5-95b5-
fdb5825c0f3d created
```

```
12:56:24,585  INFO || - [GameLiftHttpHandler] Thread-12 - API to use:
 GameLift.DescribeGameSessions
12:56:24,585  INFO || - [DescribeGameSessionsDispatcher] Thread-12 - Received API call
 to describe game sessions with input: {"FleetId":"fleet-123"}
12:56:24,660  INFO || - [GameLiftSdkHttpHandler] GameLiftSdkHttpHandler-thread-0 -
 GameLift API to use: GameSessionActivate for pId 20728
12:56:24,661  INFO || - [GameSessionActivateHandler] GameLiftSdkHttpHandler-thread-0 -
 Received API call for GameSessionActivate from 20728
12:56:24,678  INFO || - [GameSessionActivateHandler] GameLiftSdkHttpHandler-thread-0
 - GameSessionActivate data: gameSessionId: "arn:aws:gamelift:local::gamesession/
fleet-123/gsess-59f6cc44-4361-42f5-95b5-fdb5825c0f3d"
```

Matikan proses server

> **Note**
>
> Topik ini mengacu pada GameLift plugin Amazon 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.

Mengintegrasikan game dengan GameLift plugin Amazon untuk Unreal Engine

Topik di bagian ini menjelaskan GameLift plugin Amazon untuk Unreal Engine (UE) dan cara menggunakannya untuk mempersiapkan proyek game multipemain Anda untuk hosting dengan Amazon. GameLift Bekerja sepenuhnya di lingkungan pengembangan UE Anda dengan alur kerja terpandu plugin untuk melengkapi persyaratan dasar hosting dengan Amazon GameLift.

Amazon GameLift adalah layanan yang dikelola sepenuhnya yang memungkinkan pengembang game mengelola dan menskalakan server game khusus untuk game multipemain berbasis sesi. Untuk informasi selengkapnya tentang GameLift hosting Amazon, lihatCara GameLift kerja Amazon.

Topik

Tentang plugin

Plugin ini menambahkan GameLift alat dan fungsionalitas Amazon ke editor UE. Alur kerja terpandu plugin untuk mengintegrasikan Amazon GameLift ke dalam proyek game Anda, menetapkan workstation sebagai host lokal untuk pengujian, dan menyebarkan server game ke hosting cloud Amazon. GameLift

Gunakan solusi hosting bawaan plugin untuk menyebarkan game Anda. Siapkan armada Amazon GameLift Anywhere dengan workstation lokal Anda sebagai host. Untuk cloud hosting, pilih dari dua skenario penerapan umum yang menyeimbangkan latensi pemain, ketersediaan sesi game, dan biaya dengan cara yang berbeda. Satu skenario termasuk mak FlexMatch comblang sederhana dan set aturan. Gunakan solusi ini untuk memulai dengan cepat dengan struktur hosting siap produksi, lalu optimalkan dan sesuaikan sesuai kebutuhan.

Plugin ini mencakup komponen-komponen ini:

- Modul plugin untuk editor UE. Ketika plugin diinstal, tombol menu utama baru memberi Anda akses ke GameLift fungsionalitas Amazon.
- Pustaka C++ untuk API GameLift layanan Amazon dengan fungsionalitas sisi klien.
- Pustaka tidak nyata untuk SDK GameLift server Amazon (versi 5).
- 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.

Alur kerja plugin

Langkah-langkah berikut menjelaskan pendekatan khas untuk mengintegrasikan dan menerapkan proyek game dengan GameLift plugin Amazon untuk Unreal Engine. Anda menyelesaikan langkah-langkah ini dengan bekerja di editor UE dan kode game Anda.

1. Buat profil pengguna yang menautkan ke AWS akun Anda dan memberikan kredensi akses untuk pengguna akun yang valid dengan izin untuk menggunakan Amazon. GameLift
2. Tambahkan kode server ke proyek game Anda untuk membangun komunikasi antara server game yang sedang berjalan dan GameLift layanan With Amazon.
3. Tambahkan kode klien ke proyek game Anda yang memungkinkan klien game mengirim permintaan ke Amazon GameLift untuk memulai sesi game baru dan kemudian terhubung ke mereka.
4. Gunakan alur kerja Anywhere untuk menyiapkan workstation lokal Anda sebagai host Anywhere untuk server game Anda. Luncurkan server game dan klien Anda secara lokal melalui plugin, sambungkan ke sesi game, dan uji integrasi Anda.
5. Gunakan alur kerja hosting EC2 untuk mengunggah server game terintegrasi Anda dan menerapkan solusi hosting cloud, Saat server game Anda siap, luncurkan klien game Anda secara lokal melalui plugin, sambungkan ke sesi game, dan mainkan game.

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 mengenalAWS, tindakan ini mungkin tercakup dalam Tingkat AWS Gratis.

Instal plugin untuk Unreal

Bagian ini menjelaskan tugas instalasi awal untuk menambahkan plugin ke proyek Unreal Engine. Fungsionalitas plugin tersedia ketika Anda membuka proyek di editor Unreal.

> **Note**
>
> Anda dapat menggunakan GameLift plugin Amazon dengan versi standar editor UE, tetapi Anda harus menggunakan versi yang dibuat sumber saat mengemas build server game Anda.

Sebelum Anda mulai

Inilah yang Anda butuhkan untuk menggunakan GameLift plugin Amazon untuk Unreal Engine:

- GameLift Plugin Amazon untuk paket rilis Unreal Engine. [Unduh situs].
- Microsoft Visual Studio 2019 atau yang lebih baru.
- Versi sumber yang dibuat dari editor Unreal Engine. Anda memerlukan versi yang dibuat sumber untuk mengemas komponen server untuk game multipemain. Untuk detail selengkapnya, termasuk prasyarat tambahan, lihat dokumentasi Unreal Engine:
  - Mengakses kode sumber Unreal Engine di GitHub Anda akan membutuhkan GitHub dan akun Epic Games.
  - Membangun Unreal Engine dari tutorial Sumber.
- Proyek game multipemain dengan kode game C ++. Jika Anda bekerja dengan proyek Blueprint, lihat Dokumentasi tidak nyata tentang cara membuat kode sumber C++ untuk proyek Anda.

Tambahkan plugin ke proyek game Anda

Selesaikan tugas-tugas berikut untuk menambahkan plugin ke proyek game Anda.

Membangun GameLift SDK server Amazon C++

1. Buka zip GameLift plugin Amazon untuk paket rilis Unreal Engine untuk mengekstrak dua file zip:

   - `amazon-gamelift-plugin-unreal-<>-sdk-<>.zip`
   - `GameLift-Cpp-ServerSDK-<>.zip.`

   Unzip file-file ini.
2. Buka `GameLift-Cpp-ServerSDK-<>` folder, lalu lengkapi instruksi berikut untuk platform Anda: Linux atau Microsoft Windows.

Linux

1. Jalankan perintah berikut:

```
mkdir out
cd out
cmake -DBUILD_FOR_UNREAL=1 ..
```

```
make
```

Perintah ini membangun `/lib/aws-cpp-sdk-gamelift-server.so` file.

2. Salin `/lib/aws-cpp-sdk-gamelift-server.so` ke `amazon-gamelift-plugin-unreal/GameLiftPlugin/Source/GameliftServer/ThirdParty/GameLiftServerSDK/Linux/x86_64-unknown-linux-gnu/` direktori.

Microsoft Windows

1. Jalankan perintah berikut:

```
mkdir out
cd out
cmake -G "Visual Studio 17 2022" -DBUILD_FOR_UNREAL=1 ..
msbuild ALL_BUILD.vcxproj /p:Configuration=Release
```

Perintah ini membangun file biner berikut.

- `prefix\bin\aws-cpp-sdk-gamelift-server.dll`
- `prefix\lib\aws-cpp-sdk-gamelift-server.lib`

2. Salin file ke `amazon-gamelift-plugin-unreal\GameLiftPlugin\Source\GameliftServer\ThirdParty\GameLiftServerSDK\Win64\` direktori.

Selesaikan tugas-tugas berikut, kerjakan file proyek game Anda.

1. Instal file plugin.

   a. Temukan folder root proyek game Anda, seperti`... > Unreal Projects/[project-name]/`. Jika folder Plugins tidak ada di sana, maka buatlah.

   b. Buka `amazon-gamelift-plugin-unreal` folder yang dibuka ritsletingnya. `amazon-gamelift-plugin-unreal-<>-sdk-<>.zip` Salin `GameLiftPlugin` folder dari `gamelift-plugin-unreal` folder ke `Plugins` folder di direktori proyek game.

2. Tambahkan plugin ke **`.uproject`** file.

   a. Di folder root proyek game Anda, buka `.uproject` file.

b. Perbarui file untuk menambahkan GameLiftPlugin "" dan WebBrowserWidget "" ke `Plugins` bagian dan aktifkan. Kode berikut menunjukkan `.uproject` file yang diperbarui untuk game yang disebut "MyGame".

```
UnrealProjects > MyGame > MyGame.uproject
{
  ...
  "Plugins": [
    {
      "Name": "ModelingToolsEditorMode",
      "Enabled": true,
      "TargetAllowList": [ "Editor" ]
    },
    {
      "Name": "GameLiftPlugin",
      "Enabled": true
    },
    {
      "Name": "WebBrowserWidget",
      "Enabled": true
    }
  ]
}
```

3. Ubah versi editor UE untuk proyek Anda.

   Jika Anda membuat proyek untuk satu versi editor dan sekarang ingin mengubah ke versi lain (seperti versi source-build), Anda perlu memperbarui proyek.

   Di folder root proyek game Anda, pilih `.uproject` file dan pilih opsi Switch Unreal Engine Version. Pilih versi editor baru.

4. Bangun kembali solusi proyek dengan pembaruan Anda.

   a. Di folder root proyek, cari file solution (`*.sln`). Jika tidak ada, pilih `.uproject` file dan pilih opsi Hasilkan file proyek Visual Studio.

   b. Buka file solusi dan bangun atau bangun kembali proyek.

5. Verifikasi bahwa plugin diaktifkan di editor UE.

> **Note**
>
> Jika Anda Jika Anda sudah membuka editor, Anda mungkin perlu me-restart editor sebelum mengenali plugin baru.

a. Buka proyek di editor UE pilihan Anda.

b. Periksa bilah alat editor utama untuk tombol GameLift menu Amazon baru [butuh gambar].

c. Lihat di Browser Konten untuk aset GameLift plugin Amazon. Pastikan bahwa pengaturan Opsi Tampilan Anda memiliki opsi Tampilkan Konten Plugin yang dipilih.

Menyiapkan profil AWS pengguna

Setelah menginstal plugin, atur profil dan tautkan ke pengguna AWS akun yang valid. Anda dapat mempertahankan beberapa profil, tetapi Anda hanya dapat memiliki satu profil aktif pada satu waktu. Setiap kali Anda bekerja di plugin, pilih profil yang akan digunakan.

Mempertahankan beberapa profil memberi Anda kemampuan untuk beralih di antara skenario hosting yang berbeda. Misalnya, Anda dapat mengatur profil dengan AWS kredensi yang sama tetapi Wilayah yang berbedaAWS. Atau Anda dapat mengatur profil dengan AWS akun yang berbeda atau dengan pengguna/set izin yang berbeda.

> **Note**
>
> Jika Anda telah menginstal AWS CLI di workstation Anda dan memiliki profil yang sudah dikonfigurasi, GameLift plugin Amazon dapat mendeteksinya dan akan 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.

Untuk mengelola AWS profil Anda

1. Di toolbar utama editor Unreal, pilih GameLift menu Amazon, dan pilih Set AWS User Profiles. Tindakan ini membuka Pengaturan Proyek untuk plugin. Perluas bagian Profil AWS Pengguna.

2. Jika plugin tidak mendeteksi profil yang ada, itu meminta Anda untuk membuatnya. Anda dapat membuat profil baru menggunakan AWS akun baru atau yang sudah ada.

   > **Note**
   >
   > Anda perlu menggunakan Konsol AWS Manajemen untuk membuat AWS akun baru dan membuat atau memperbarui pengguna dengan set izin yang tepat.

   Saat mengatur profil, Anda memerlukan informasi berikut:

   - Akun AWS Jika Anda perlu membuat AWS akun baru, ikuti petunjuk untuk membuat akun. Lihat Membuat AWS akun untuk detail selengkapnya.
   - AWSPengguna dengan izin untuk menggunakan Amazon GameLift dan AWS layanan lain yang diperlukan. Lihat Siapkan Akun AWS petunjuk tentang cara menyiapkan pengguna AWS Identity and Access Management (IAM) dengan GameLift izin Amazon dan akses terprogram dengan kredensi jangka panjang.
   - Kredensil untuk pengguna AndaAWS. Kredensyal ini terdiri dari ID kunci AWS akses dan kunci AWS rahasia. Lihat Dapatkan kunci akses Anda untuk detail selengkapnya.
   - AWSwilayah. 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. Lihat daftar AWSwilayah yang didukung.

3. Jika plugin mendeteksi profil yang ada, Anda tidak diminta untuk membuatnya. Jika Anda ingin memperbarui profil atau membuat yang baru, pilih Kelola profil Anda.

Untuk bootstrap profil Anda:

Semua profil harus di-bootstrap untuk digunakan dengan plugin Amazon. GameLift 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.

1. Di toolbar utama editor Unreal, pilih GameLift ikon Amazon, dan pilih Set AWS User Profiles. Tindakan ini membuka Pengaturan Proyek untuk plugin. Perluas bagian Profil AWS Pengguna.

2. Di bagian Bootstrap profil Anda, pilih profil dari daftar dropdown dan periksa status bootstrap. Jika status menunjukkan bahwa tidak ada bucket, pilih tombol Bootstrap dan buat profil untuk membuat bucket Amazon S3 untuk profil yang dipilih.
3. Tunggu status bootstrap berubah menjadi “Aktif”. Hal ini dapat menghabiskan waktu beberapa menit.

Siapkan game Anda untuk pengujian dengan Amazon GameLift Anywhere

Dalam alur kerja ini, Anda menambahkan kode game klien dan server untuk GameLift fungsionalitas Amazon, 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 Anywhere:

- Di bilah alat utama editor Unreal, pilih GameLift menu Amazon, 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.

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, Anda membuat serangkaian pembaruan untuk klien dan kode server Anda untuk menambahkan fungsionalitas hosting. Jika Anda belum menyiapkan versi editor Unreal yang dibuat sumber, plugin menyediakan tautan ke instruksi dan kode sumber.

Dengan plugin, dapat memanfaatkan beberapa kemudahan saat mengintegrasikan kode game Anda. Anda dapat melakukan integrasi minimal untuk mengatur fungsionalitas hosting dasar. Anda juga dapat melakukan integrasi kustom yang lebih luas. Informasi di bagian ini menjelaskan opsi integrasi minimal. Gunakan peta pengujian yang disertakan dengan plugin untuk menambahkan GameLift fungsionalitas Amazon klien ke proyek game Anda. Untuk integrasi server, gunakan contoh kode yang disediakan untuk memperbarui mode permainan proyek Anda.

Integrasikan mode permainan server Anda

Tambahkan kode server ke game Anda yang memungkinkan komunikasi antara server game Anda dan GameLift layanan Amazon. Server game Anda harus dapat menanggapi permintaan dari Amazon GameLift, seperti memulai sesi permainan baru, dan juga melaporkan status kesehatan server game dan koneksi pemain.

1. Di editor kode Anda, buka file solution (`.sln`) untuk proyek game Anda, biasanya ditemukan di folder root proyek. Misalnya: `GameLiftUnrealApp.sln`.
2. Dengan solusi terbuka, cari file header mode game proyek: `[project-name]GameMode.h` file. Misalnya: `GameLiftUnrealAppGameMode.h`.
3. Ubah file header agar sejajar dengan kode contoh 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.

```
// 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 "GameLiftServerGameMode.generated.h"

struct FProcessParameters;

DECLARE_LOG_CATEGORY_EXTERN(GameServerLog, Log, All);

UCLASS(minimalapi)
class AGameLiftServerGameMode : public AGameModeBase
{
    GENERATED_BODY()
```

```
public:
    AGameLiftServerGameMode();

protected:
    virtual void BeginPlay() override;

private:
    void InitGameLift();

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

4. Buka file `[project-name]GameMode.cpp` file sumber terkait (misalnya`GameLiftUnrealAppGameMode.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 dengan Amazon GameLift:

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

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

#include "GameLiftServerGameMode.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);

AGameLiftServerGameMode::AGameLiftServerGameMode() :
    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 AGameLiftServerGameMode..."));
}

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

#if WITH_GAMELIFT
    InitGameLift();
#endif
}

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

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

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

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

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

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

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

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

```
            ServerParametersForAnywhere.m_processId =
                TCHAR_TO_UTF8(
                    *FText::Format(
                        FText::FromString("ProcessId_{0}"),
                        FText::AsNumber(std::time(nullptr))
                    ).ToString()
                );
        }

        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);
        }

        UE_LOG(GameServerLog, SetColor, TEXT("%s"), COLOR_YELLOW);
        UE_LOG(GameServerLog, Log, TEXT(">>>> Web Socket 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, 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, GameLift 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. GameLift 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 GameLift 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.
    //GameLift 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. GameLift
will default to 'false' if the game server doesn't respond in time.
    //In this case, we're always healthy!
    ProcessParameters->OnHealthCheck.BindLambda([]()
    {
        UE_LOG(GameServerLog, Log, TEXT("Performing Health Check"));
        return true;
    });

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

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

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

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

    //Here, the game server tells GameLift where to find game session log files.
    //At the end of a game session, GameLift 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 GameLift 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 armada EC2 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 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.

Bangun komponen game Anda

1. Buat file target server dan klien baru

    a. Di folder proyek game Anda, buka folder Sumber dan temukan `Target.cs` file.

    b. Salin file `[project-name]Editor.Target.cs` ke dua file baru bernama `[project-name]Client.Target.cs` dan`[project-name]Server.Target.cs`.

    c. Edit setiap file baru untuk memperbarui nama kelas dan nilai tipe target, seperti yang ditunjukkan:

```
UnrealProjects > MyGame > Source > MyGameClient.Target.cs
```

```
// Copyright Epic Games, Inc. All Rights Reserved.

using UnrealBuildTool;
using System.Collections.Generic;

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

```
UnrealProjects > MyGame > Source > MyGameServer.Target.cs
// Copyright Epic Games, Inc. All Rights Reserved.

using UnrealBuildTool;
using System.Collections.Generic;

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

2. Perbarui `.Build.cs` file.

   a. Buka `.Build.cs` file untuk proyek Anda. File ini terletak di `UnrealProjects/[project name]/Source/[project name]/[project name].Build.cs`.

   b. Perbarui `ModuleRules` kelas seperti yang ditunjukkan pada contoh kode berikut.

```
public class MyGame : ModuleRules
{
  public GameLiftUnrealApp(TargetInfo Target)
```

```
  {
    PublicDependencyModuleNames.AddRange(new string[] { "Core", "CoreUObject",
 "Engine", "InputCore" });
    bEnableExceptions = true;

    if (Target.Type == TargetRules.TargetType.Server)
    {
      PublicDependencyModuleNames.AddRange(new string[]
 { "GameLiftServerSDK" });
      PublicDefinitions.Add("WITH_GAMELIFT=1");
    }
    else
    {
      PublicDefinitions.Add("WITH_GAMELIFT=0");
    }
  }
}
```

3. Membangun kembali solusi proyek game Anda.
4. Buka proyek game Anda dalam versi editor Unreal Engine yang dibuat sumber.
5. Lakukan hal berikut untuk klien dan server Anda:

   a. Pilih target. Buka Platform, Windows dan pilih salah satu dari yang berikut ini:

      - Server: `[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.

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 memverifikasi bahwa status armada adalah iklan aktif 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.

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 memverifikasi bahwa ia 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

Langkah 5: Hasilkan token autentikasi

Proses server game yang berjalan pada komputasi Anywhere Anda memerlukan token otentikasi untuk melakukan panggilan ke layanan. GameLift 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.

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

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

Terapkan game Anda ke cloud hosting dengan armada EC2 terkelola

Dalam alur kerja ini, Anda menggunakan plugin untuk memodifikasi game untuk dihosting di sumber daya komputasi berbasis cloud yang dikelola oleh Amazon. GameLift Anda menambahkan kode game klien dan server untuk GameLift fungsionalitas Amazon, lalu mengunggah build server Anda ke GameLift layanan Amazon untuk diterapkan ke sumber daya berbasis cloud. Ketika alur kerja ini selesai, Anda akan memiliki klien game yang berfungsi yang dapat terhubung ke server game Anda di cloud.

Untuk memulai alur kerja Amazon EC2 yang GameLift dikelola Amazon:

- Di toolbar utama editor Unreal, pilih GameLift menu Amazon, 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.

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, Anda membuat serangkaian pembaruan untuk klien dan kode server Anda untuk menambahkan fungsionalitas hosting. Jika Anda belum menyiapkan versi editor Unreal yang dibuat sumber, plugin menyediakan tautan ke instruksi dan kode sumber.

Jika Anda mengintegrasikan game Anda untuk digunakan dengan armada Anywhere, Anda tidak perlu membuat perubahan apa pun pada kode game Anda. Jika Anda menggunakan peta game startup, ini juga berfungsi dengan penerapan EC2.

- Siapkan kode game Anda (Di Mana Saja)
- Bangun komponen game Anda

Setelah membangun server game Anda, selesaikan tugas-tugas berikut untuk mempersiapkannya untuk diunggah ke Amazon GameLift.

Untuk mengemas build server Anda untuk penerapan cloud

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

1. Salin skrip instalasi, termasuk dalam unduhan plugin, ke root `WindowsServer` folder. Cari filenya`[project-name]/Plugins/Resources/CloudFormation/extra_server_resources/install.bat`. Amazon GameLift menggunakan file ini untuk menginstal build server pada setiap sumber daya hosting EC2.
2. Salin `VC_redist.x64.exe` file, termasuk dalam instalasi Visual Studio Anda, ke root `WindowsServer` folder. File ini biasanya terletak di`C:/Program Files (x86)/Microsoft Visual Studio/2019/Professional/VC/Redist/MSVC/v142`.
3. Salin DLL OpenSSL untuk server game Anda yang dibangun ke dalam folder. `WindowsServer/MyGame/Binaries/Win64` Pastikan DLL untuk versi yang sama yang digunakan dalam pembuatan server. Salin file-file berikut:
   - `libssl-3-x64.dll`
   - `libcrypto-3-x64.dll`

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:
  - AWSarmada (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 API.
  - WebACL 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.
  - AWSantrian 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 API.

- WebACL 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. AWS

- Nama pembuatan server: Berikan nama yang berarti untuk pembuatan server game Anda. AWSmenggunakan 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 selama 40 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 instans AWS EC2. 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 yang dihosting dengan Amazon GameLift. 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.

Mendapatkan data armada untuk instans Amazon GameLift

Ada beberapa situasi di mana pembuatan game kustom Anda atau skrip Server Realtime mungkin memerlukan informasi tentang GameLift armada Amazon. 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 aktivitas.)
- 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.json`File tersebut menyertakan atribut sumber daya GameLift armada Amazon.

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",
    "fleetName":"ReallyFunGameTestFleet08",
    "fleetType":"ON_DEMAND",
    "instanceRoleArn":"arn:aws:iam::123456789012:role/S3AccessForGameLift",
    "instanceType":"c5.large",
    "serverLaunchPath":"/local/game/reallyfungame.exe"
}
```

Menambahkan FlexMatch perjodohan

Gunakan Amazon GameLift FlexMatch untuk menambahkan fungsionalitas perjodohan pemain ke game yang GameLift dihosting Amazon Anda. Anda dapat menggunakan FlexMatch dengan server game khusus atau Server Realtime.

FlexMatchmemasangkan layanan perjodohan dengan mesin aturan yang dapat disesuaikan. Anda merancang cara matchmaking pemain bersama-sama berdasarkan atribut pemain dan mode game yang masuk akal untuk game 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 lengkap, Anda harus menyiapkan sumber daya hosting Anda dengan antrian. Amazon GameLift menggunakan antrean untuk menemukan lokasi hosting terbaik untuk game di berbagai wilayah dan jenis komputasi. Secara khusus, GameLift antrian Amazon dapat menggunakan data latensi, ketika disediakan oleh klien game, untuk menempatkan sesi game sehingga pemain mengalami latensi serendah mungkin saat bermain.

Untuk informasi selengkapnya tentang FlexMatch menyertakan bantuan terperinci dengan mengintegrasikan perjodohan ke dalam game Anda, lihat topik Panduan GameLiftFlexMatchPengembang Amazon ini:

- Cara GameLift FlexMatch kerja Amazon

- FlexMatchlangkah integrasi

Mengelola hosting dengan GameLift wadah Amazon

Amazon GameLift menyediakan layanan cloud hosting lengkap untuk mendukung solusi kontainer untuk hosting server game. Dengan armada GameLift kontainer Amazon, Anda dapat memanfaatkan manfaat kontainer seperti portabilitas, kelincahan, dan toleransi kesalahan.

Fitur utama

Fitur-fitur berikut tersedia dengan armada GameLift kontainer Amazon.

- Kembangkan arsitektur kontainer khusus dengan kontainer ringan untuk menjalankan perangkat lunak server game Anda di sumber daya GameLift hosting Amazon.
- Sertakan GameLift Agen Amazon untuk mengelola siklus hidup proses server game di dalam container Anda. Agen on-compute melaksanakan instruksi Anda tentang kapan dan bagaimana memulai proses server dan berapa banyak yang harus dipertahankan untuk hosting sesi game.
- Sesuaikan sumber daya yang disediakan oleh Amazon GameLift untuk membuat gambar kontainer dengan aplikasi server game Anda. Gunakan dockerfile yang disediakan untuk membuat gambar kontainer berbasis Linux. Simpan gambar untuk armada kontainer Anda di repositori pribadi Amazon Elastic Container Registry (Amazon ECR).
- Memberikan pengalaman pemutar latensi rendah dengan menerapkan sumber daya armada kontainer ke Zona Lokal mana Wilayah AWS pun yang didukung Amazon. GameLift Buat armada kontainer multi-lokasi untuk manajemen armada yang efisien. Lihat Lokasi GameLift hosting Amazon.
- Uji solusi hosting game kontainer Anda dengan armada Amazon GameLift Anywhere. Gunakan Anywhere untuk menguji pengembangan solusi secara lokal, termasuk integrasi Amazon GameLift SDK dan konfigurasi image container Anda.
- Lacak kinerja hosting game dengan metrik kinerja khusus container. Pantau kesehatan sumber daya armada Anda menggunakan metrik perangkat keras.
- Gunakan alat penempatan sesi GameLift permainan Amazon, termasuk antrian dan FlexMatch perjodohan, untuk mencocokkan pemain dengan sesi permainan terbaik yang diselenggarakan di armada kontainer Anda.
- Kelola sumber daya armada kontainer menggunakan AWS CloudFormation templat untuk Amazon GameLift.

Menggunakan armada kontainer selama pratinjau publik

Fitur armada kontainer baru saat ini dalam pratinjau publik. Selama fase ini, GameLift fitur Amazon berikut didukung:

- Gunakan armada kontainer untuk meng-host server game yang dibangun untuk Linux. Armada kontainer menggunakan `Amazon_Linux_2023` dan mendukung gambar kontainer Linux. Kontainer Windows tidak didukung.
- Integrasikan proyek server game dengan Amazon GameLift server SDK versi 5+ saja. Versi sebelumnya tidak didukung.
- Gunakan salah satu jenis instans On-Demand Amazon EC2 yang didukung Amazon. GameLift Armada spot tidak didukung saat ini.

Cara kerja kontainer di Amazon GameLift

Armada GameLift kontainer Amazon dirancang untuk memberikan 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 Amazon Anda. GameLift Topik ini menjelaskan elemen struktural dasar untuk menjalankan kontainer pada armada yang GameLift dikelola Amazon, menggambarkan arsitektur umum, dan menguraikan beberapa konsep inti.

Komponen armada kontainer

Armada

Armada kontainer adalah kumpulan instans Amazon EC2, yang dikelola oleh Amazon GameLift, yang menjalankan server game kontainer Anda. Saat membuat armada, Anda mengonfigurasi bagaimana arsitektur kontainer dan perangkat lunak server game Anda diterapkan ke setiap instance armada. Anda dapat menyebarkan armada kontainer ke satu Wilayah AWS atau beberapa lokasi geografis. Anda dapat menggunakan alat penskalaan GameLift manual atau otomatis Amazon untuk menskalakan kapasitas armada kontainer untuk menyelenggarakan sesi permainan dan pemain.

Instans

Instans Amazon EC2 adalah server virtual yang menyediakan kapasitas komputasi untuk hosting game Anda. Dengan Amazon GameLift, 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 akan menerapkan instance 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 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 cara kontainer berbagi sumber daya CPU dan memori yang tersedia. Anda juga dapat mengatur dependensi antar kontainer dan mengelola siklus hidup grup kontainer.

Grup kontainer dapat mereplikasi di setiap instance armada untuk mengoptimalkan penggunaan sumber daya. Anda dapat mengelola replikasi dengan menetapkan strategi penjadwalan grup kontainer, sebagai berikut:

- Grup kontainer replika mengelola kontainer yang menjalankan aplikasi server game Anda dan perangkat lunak pendukung. Semua armada kontainer harus mendefinisikan grup kontainer replika. Grup replika mungkin memiliki beberapa salinan pada setiap instance armada, tergantung pada persyaratan grup kontainer dan sumber daya dari jenis instance yang digunakan. Semua kontainer dalam grup replika secara otomatis menskalakan bersama di seluruh instance.
- Grup kontainer daemon, yang bersifat opsional, mungkin 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 daemon. Grup kontainer daemon tidak direplikasi - setiap instance armada memiliki paling banyak satu salinan grup daemon. Ini berarti bahwa kontainer dalam grup daemon tidak menskalakan seluruh instance armada bersama dengan kontainer dalam grup replika.

Armada kontainer harus memiliki satu grup kontainer replika dan secara opsional dapat memiliki satu grup daemon.

Kontainer

Wadah adalah elemen paling dasar dari arsitektur berbasis kontainer. Ini terdiri dari gambar kontainer dengan perangkat lunak yang dapat dieksekusi dan file dependen. Saat Anda

menentukan wadah yang akan digunakan dengan Amazon GameLift, Anda mengonfigurasi cara perangkat lunak berjalan di penampung.

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

Jenis kontainer meliputi:

- Wadah replika penting mencakup semua yang Anda butuhkan untuk menjalankan proses server game dan menyelenggarakan sesi permainan untuk pemain. Ini termasuk build server game Anda, yang terintegrasi dengan SDK GameLift server Amazon, dan perangkat lunak dependen. Ini juga termasuk GameLift Agen Amazon, yang mengelola siklus hidup proses server game Anda. Kelompok kontainer replika armada memiliki persis satu wadah replika penting.
- Kontainer replika non-esensial, juga disebut wadah "sidecar", menjalankan perangkat lunak untuk mendukung aplikasi server game Anda. Menggunakan wadah sespan memungkinkan Anda menjalankan dan menskalakan perangkat lunak pendukung di samping server game Anda tetapi mengelola sebagai wadah terpisah. Jika jenis penampung ini gagal, hanya penampung itu sendiri yang memulai ulang; grup kontainer tidak terpengaruh.
- Kontainer daemon menjalankan layanan daemon untuk mengelola proses latar belakang. Penggunaan umum untuk wadah daemon adalah menjalankan agen Amazon CloudWatch (CloudWatch) untuk mengumpulkan metrik, log, dan jejak untuk kontainer Anda. Wadah daemon dapat menjadi penting atau tidak penting tergantung pada kapan kegagalan kontainer harus mengakibatkan restart grup kontainer.

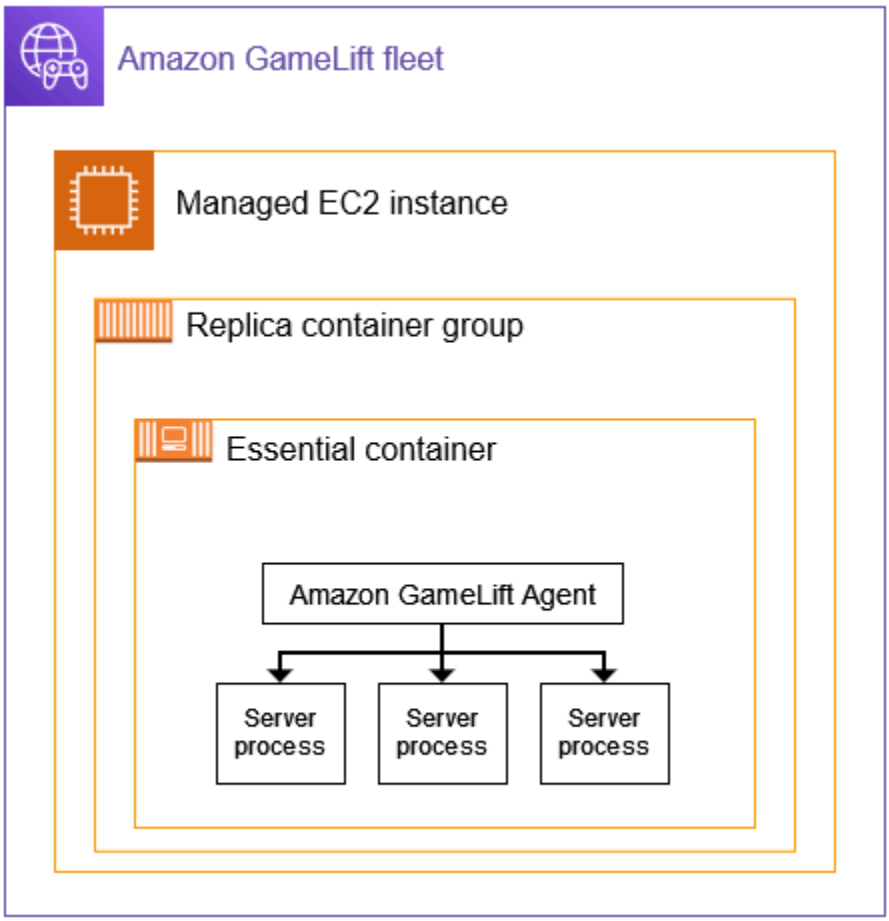
Hitung

Compute adalah sumber daya hosting armada yang terdaftar di GameLift layanan Amazon dan dapat berkomunikasi dengan layanan tersebut. Dalam armada kontainer, komputasi adalah wadah dengan proses yang mengelola proses pendaftaran komputasi. Dalam wadah replika penting armada kontainer, GameLift Agen Amazon secara otomatis mendaftarkan wadah ini sebagai komputasi.

Arsitektur umum

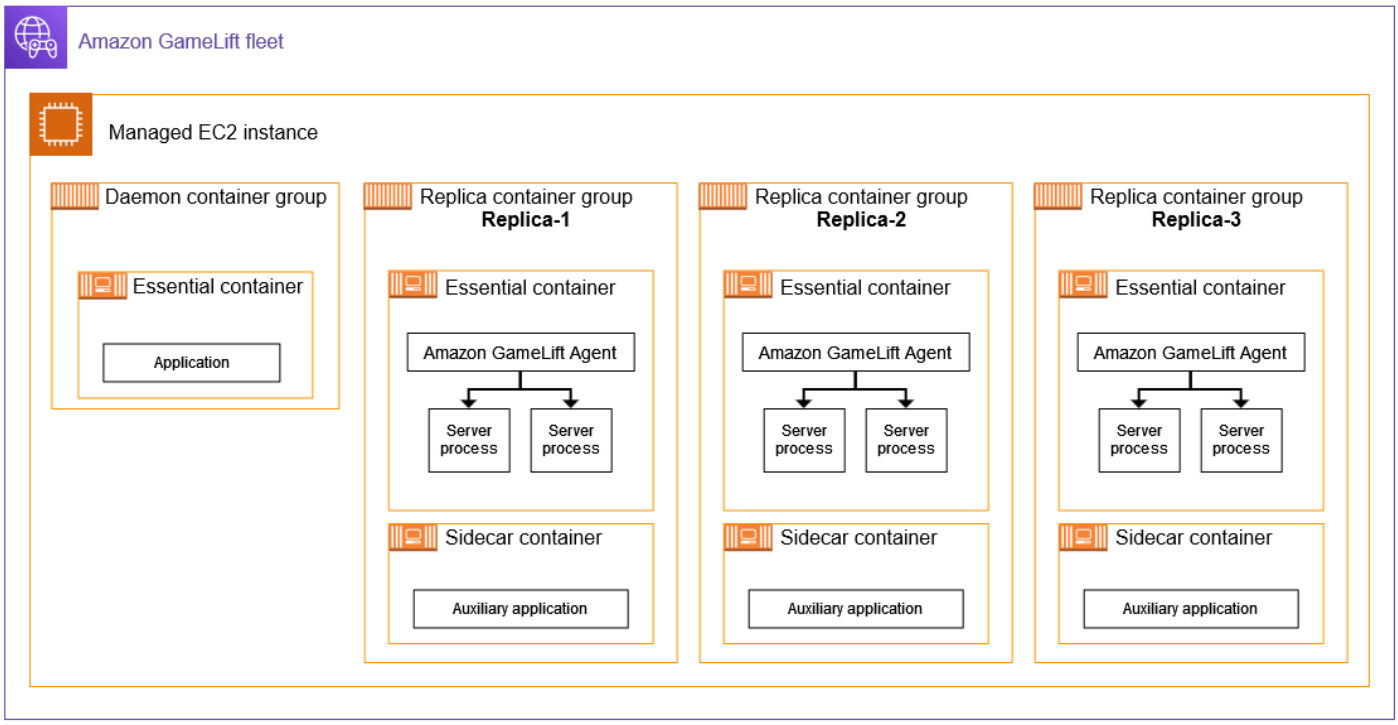
Diagram berikut menggambarkan struktur armada kontainer yang paling sederhana. Dalam struktur ini, setiap instance dalam armada memelihara satu salinan grup kontainer replika. Grup kontainer

memiliki satu wadah penting yang menjalankan GameLift Agen Amazon, aplikasi server game, dan semua perangkat lunak pendukung untuk hosting sesi game. Agen mengimplementasikan instruksi khusus armada untuk menjalankan tiga proses server secara bersamaan. Karena ada satu grup kontainer replika per instance, setiap instance armada menjalankan tiga proses server secara bersamaan.



Contoh kedua ini menggambarkan desain armada kontainer yang lebih kompleks. Dalam contoh ini, armada memiliki grup kontainer replika dengan beberapa kontainer dan grup kontainer daemon dengan satu kontainer. Konfigurasi armada menempatkan tiga salinan grup kontainer replika pada setiap instance armada. Grup wadah daemon tidak pernah direplikasi.

Setiap set wadah grup replika di memiliki tiga salinan pada setiap instance. Dalam setiap wadah replika penting, Agen diinstruksikan untuk menjalankan dua proses server secara bersamaan. Akibatnya, setiap instance armada menjalankan enam proses server secara bersamaan (dua proses di masing-masing dari tiga wadah replika penting).



Konsep inti

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

Pengepakan kelompok kontainer

Saat mengembangkan struktur kontainer Anda untuk penyebaran dalam armada kontainer, tujuan umum adalah mengoptimalkan penggunaan daya komputasi yang tersedia. Untuk mencapai tujuan ini, temukan jumlah grup kontainer replika tertinggi yang dapat Anda tempatkan pada instance armada tanpa memengaruhi kinerja server game.

Amazon GameLift dapat membantu Anda melakukan ini. Ini menghitung jumlah maksimum kelompok replika per instance, berdasarkan informasi berikut:

- Jenis instance armada dan CPU dan sumber daya memori yang tersedia.
- Persyaratan CPU dan memori yang Anda tetapkan untuk semua kontainer dalam grup replika Anda.

  Persyaratan CPU dan memori yang Anda tetapkan untuk semua kontainer dalam grup daemon, jika ada.

- Sumber daya disediakan untuk mengelola kontainer dan aplikasi penting lainnya pada setiap instance.

Saat membuat armada kontainer, Anda dapat memilih untuk menggunakan jumlah maksimum yang dihitung atau Anda dapat mengganti nomor yang dihitung dengan menentukan nomor yang diinginkan. Sebagai praktik terbaik, bereksperimenlah dengan perangkat lunak server game kontainer Anda untuk menentukan kebutuhan sumber daya yang akurat. Gunakan data ini untuk menemukan strategi pengemasan yang optimal untuk kinerja server game.

Server game dan GameLift Agen Amazon

Saat Anda membangun wadah replika penting Anda, Anda mengemas perangkat lunak server game Anda dan GameLift Agen Amazon bersama-sama dalam gambar kontainer yang sama. Agen on-compute ini mengontrol siklus hidup server game dalam container. Di setiap grup kontainer replika, wadah replika penting menjalankan Agen dan semua proses server game.

GameLift Agen Amazon mengeksekusi instruksi dalam konfigurasi runtime armada kontainer. Konfigurasi runtime mengidentifikasi (1) executable untuk mulai berjalan, (2) set opsional parameter peluncuran, dan (3) jumlah proses yang akan dijalankan secara bersamaan. Konfigurasi runtime dapat memiliki instruksi untuk beberapa executable yang berbeda. Setidaknya satu instruksi harus untuk server game yang dapat dieksekusi. Misalnya, konfigurasi runtime mungkin menginstruksikan agen untuk mempertahankan 10 proses server game Anda yang dapat dieksekusi untuk penggunaan produksi, 1 proses eksekusi yang sama dengan parameter peluncuran khusus untuk pengujian, dan 1 proses untuk utilitas logging.

Anda dapat memodifikasi konfigurasi runtime armada kapan saja. GameLift Agen Amazon secara berkala meminta pembaruan dari layanan. Ketika konfigurasi runtime yang diperbarui tersedia, Agen menerimanya dan mulai menerapkan instruksi. Tindakan mungkin termasuk menambahkan atau mematikan proses server.

GameLift Agen Amazon adalah versi open-source dari agen on-compute yang GameLift digunakan Amazon untuk armada EC2 terkelola. Panduan ini memberikan petunjuk tentang cara membuat Agen dari sumber dan membuatnya menjadi gambar kontainer. Agen menangani tugas-tugas berikut:

Manajemen proses server:

- Mulai, matikan, dan ganti proses server berdasarkan konfigurasi runtime.
- Matikan proses server ketika mereka tidak aktif tepat waktu.

- Laporkan ke Amazon GameLift saat proses server berakhir.
- Memancarkan peristiwa armada untuk proses server.

Manajemen kontainer:

- Matikan proses server sebagai respons terhadap permintaan dari Amazon GameLift.
- Laporkan kesehatan kontainer.

Tugas pengunggahan log:

- Unggah log sesi game ke bucket Amazon S3 yang ditentukan.
- Unggah log agen saat komputasi ke bucket Amazon S3 yang ditentukan.

Penskalaan kapasitas armada

Kapasitas armada mengukur jumlah sesi permainan yang dapat diselenggarakan armada pada satu waktu. Anda juga dapat mengukur kapasitas berdasarkan jumlah pemain yang dapat didukung armada secara bersamaan.

Untuk menambah atau mengurangi kapasitas hosting armada, Anda menambah atau menghapus instance armada. Strategi pengepakan armada kontainer menentukan berapa banyak sesi permainan yang berjalan bersamaan pada setiap instance armada. Angka ini memberi tahu Anda jumlah sesi permainan (dan slot pemain) yang Anda tambahkan atau kurangi saat Anda menambah atau mengurangi kapasitas armada.

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

- Atur kapasitas armada secara manual dengan menetapkan jumlah instans armada tertentu yang diinginkan.
- Siapkan penskalaan otomatis dengan menargetkan buffer yang diinginkan dari instance yang tersedia (pelacakan target). Metode ini secara otomatis mempertahankan satu set sumber daya hosting yang menganggur sehingga pemain yang masuk selalu 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).

Koneksi klien/server game

Armada EC2 terkelola dan armada kontainer menangani koneksi antara klien game dan server game yang dihosting cloud dengan cara yang sama. Saat Amazon GameLift membuat sesi permainan baru, layanan mengkomunikasikan informasi koneksi sesi game. Klien game menggunakan informasi untuk terhubung langsung ke server game yang menjadi tuan rumah sesi game. Untuk semua jenis armada, informasi koneksi terdiri dari alamat IP dan penetapan port.

Saat membuat armada kontainer, Anda menentukan dua set rentang port. Pertama, Anda menentukan berbagai port koneksi yang menghadap eksternal yang memungkinkan klien game terhubung ke game. Kedua, Anda menentukan satu set port kontainer internal saja, yang ditetapkan untuk setiap proses server game yang berjalan di wadah. Amazon GameLift secara dinamis memetakan port kontainer internal ke port koneksi eksternal untuk memberi pemain akses ke game. Pendekatan ini memberikan lapisan keamanan tambahan dengan melindungi server game Anda dari akses langsung ke port kontainer.

Saat menentukan rentang port untuk armada kontainer, Anda harus menyediakan rentang dengan port yang cukup untuk mengakomodasi semua proses server yang berjalan secara bersamaan di seluruh kontainer pada sebuah instance.

Untuk kontrol tambahan, Anda juga menetapkan izin masuk untuk armada. Izin masuk menentukan port koneksi mana yang terbuka untuk lalu lintas masuk. Anda dapat mengubah izin masuk armada kapan saja. Dengan izin masuk, Anda dapat dengan cepat mematikan semua port koneksi, membuka beberapa, atau membuka semua sesuai kebutuhan.

Peta jalan pengembangan untuk wadah Amazon GameLift

Alur kerja berikut merangkum langkah-langkah agar server game Anda berjalan di armada GameLift penampung Amazon.

Langkah 1: Integrasikan game Anda dengan Amazon GameLift

Tambahkan fungsionalitas ke server game Anda sehingga dapat berkomunikasi dengan GameLift layanan Amazon saat digunakan ke armada kontainer. Jika Anda menggunakan FlexMatch perjodohan, tambahkan fungsi ini ke server dan klien game Anda. Untuk detail informasi, lihat Integrasikan game Anda dengan Amazon GameLift.

- Dapatkan SDK GameLift server Amazon (versi 5+) dan atur dengan proyek game Anda. Server SDK tersedia dalam C ++, C #, dan Go.
- Ubah kode server game Anda untuk menambahkan fungsionalitas SDK server yang diperlukan.

- Package build server game Anda untuk Linux. Jika Anda mengembangkan di Windows, langkah ini mungkin memerlukan pekerjaan tambahan untuk menyiapkan lingkungan Linux.
- (Opsional) Uji integrasi server game Anda menggunakan GameLift Anywhere armada Amazon. Uji sebelum menyiapkan gambar kontainer Anda untuk mengisolasi masalah dengan pekerjaan integrasi Anda. Untuk menguji koneksi klien/server game, integrasikan klien game Anda juga.

> **Note**
>
> Jika Anda mengembangkan di Windows, siapkan ruang kerja Linux terpisah, atau gunakan 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.

Langkah 2: Siapkan gambar wadah server game Anda

Buat image kontainer yang menjalankan proses server game Anda dan simpan di repositori Amazon Elastic Container Registry (Amazon ECR) untuk digunakan dengan Amazon. GameLift Untuk petunjuk mendetail, lihat Siapkan gambar kontainer dengan perangkat lunak server game Anda.

- Siapkan direktori kerja untuk image container Anda, dengan build game Linux Anda, instal skrip, dan semua perangkat lunak dan dependensi pendukung.
- Dapatkan kode sumber GameLift Agen Amazon, buat, dan tambahkan `jar` file ke direktori kerja Anda.
- Dapatkan Dockerfile default dan modifikasi untuk mengkonfigurasi gambar kontainer dengan perangkat lunak server game Anda.
- Bangun gambar kontainer Anda. Lakukan langkah ini di lingkungan Linux.
- Buat repositori pribadi Amazon ECR dan dorong gambar kontainer Anda ke sana. Buat repo di tempat yang sama Akun AWS dan di Wilayah AWS mana Anda berencana untuk menyebarkan armada kontainer Anda.
- (opsional) Uji gambar kontainer Anda menggunakan Anywhere armada Anda. Anda dapat mengatur konfigurasi runtime untuk meneruskan instruksi ke GameLift Agen Amazon.

Langkah 3: Buat wadah dan grup kontainer Anda

Rancang arsitektur kontainer untuk hosting game di Amazon GameLift. Lihat Rancang armada GameLift kontainer Amazon dan Buat definisi grup kontainer untuk armada GameLift kontainer Amazon.

- Tentukan konfigurasi kontainer Anda. Untuk setiap kontainer, Anda akan menentukan masalah seperti proses runtime, alokasi memori, pemeriksaan kesehatan, port jaringan, dll.
- Gunakan GameLift konsol Amazon atau AWS CLI untuk membuat definisi grup kontainer dengan konfigurasi container Anda. Saat Anda membuat definisi grup kontainer, Amazon GameLift mengambil snapshot dari setiap gambar kontainer pada saat itu.

Langkah 4: Menyebarkan server game kontainer Anda ke armada kontainer

Gunakan definisi grup kontainer yang dibuat pada langkah sebelumnya untuk membuat armada kontainer dan menyebarkan perangkat lunak server game kontainer Anda. Lihat Buat armada GameLift kontainer Amazon.

- Gunakan GameLift konsol Amazon atau AWS CLI untuk membuat armada kontainer.
- Lacak status armada saat instance armada disebarkan dan diaktifkan. Periksa peristiwa pembuatan armada untuk memverifikasi bahwa armada berhasil dikerahkan ke semua lokasi.
- Verifikasi bahwa klien game dapat meminta dan bergabung dengan sesi permainan dan memainkan game. Jika Anda telah menyiapkan perjodohan, uji skenario tersebut.

Langkah 5: Kelola armada Anda

Saat Anda mempersiapkan penggunaan tingkat produksi, buat solusi hosting game Anda dan kelola siklus hidup hosting Anda.

- Buat armada dan armada multi-lokasi di tempat lain Wilayah AWS untuk mendukung basis pemain Anda.
- Konfigurasikan penempatan hosting game dengan antrian atau FlexMatch perjodohan. Lihat sumber daya ini:
  - Menyiapkan GameLift antrean Amazon untuk penempatan sesi game
  - FlexMatch Panduan Pengembang
- Siapkan penskalaan otomatis untuk mengelola kapasitas armada berdasarkan permintaan pemain untuk sesi permainan.
- Siapkan pemantauan untuk armada kontainer Anda. Bekerja dengan GameLift metrik Amazon, ambil log sesi game dan log kontainer, atur akses jarak jauh ke kontainer individual.
- Mengatur manajemen jangka panjang armada kontainer. Gunakan alias armada untuk merampingkan proses memperbarui armada kontainer. Buat AWS CloudFormation template untuk mengelola siklus hidup armada. Lihat sumber daya ini:
  - Tambahkan alias ke armada Amazon GameLift
  - Mengelola sumber daya menggunakan AWS CloudFormation

Integrasikan game Anda dengan Amazon GameLift

Sebelum Anda dapat membuat gambar kontainer dengan perangkat lunak server game Anda dan menyebarkannya ke Amazon GameLift untuk hosting cloud, pertama-tama integrasikan proyek game Anda dengan SDK GameLift server Amazon dan buat server game untuk dijalankan di Linux. Topik ini memperkenalkan berbagai alat integrasi yang GameLift disediakan Amazon.

Server game yang di-host harus dapat berkomunikasi dengan GameLift layanan Amazon. Siapkan komunikasi dengan menambahkan SDK GameLift server Amazon (versi 5+) ke proyek game Anda dan memodifikasi kode server game Anda. Amazon GameLift menyediakan sumber daya dan dokumentasi SDK server untuk mendukung beberapa bahasa dan mesin game.

Proses integrasi untuk server game kontainer hampir identik dengan mengintegrasikan server game untuk hosting pada armada EC2 atau Amazon yang dikelola. GameLift Anywhere

Alat integrasi

Amazon GameLift menyediakan alat dan dukungan bahasa berikut untuk integrasi:

Untuk pengembang Unreal Engine

Gunakan plugin ringan untuk Unreal. Plugin ini mencakup pustaka SDK server C ++ dengan fungsionalitas Amazon yang diperlukan. GameLift Gunakan dokumentasi untuk mengonfigurasi proyek game Unreal Anda untuk plugin dan perbarui kode game Anda dengan blok kode yang disediakan untuk menambahkan fungsionalitas yang diperlukan untuk server dan build klien Anda.

- Unduhan plugin SDK
- Panduan: Integrasikan proyek Unreal Anda dengan Amazon GameLift
- Panduan referensi: C ++ Server SDK 5 untuk Unreal

Catatan: Plugin GameLift mandiri Amazon untuk Unreal Engine tidak mendukung penggunaan armada kontainer.

Untuk pengembang Unity

Gunakan plugin ringan untuk Unity. Plugin ini mencakup pustaka SDK server C # dengan fungsionalitas Amazon yang diperlukan. GameLift Gunakan dokumentasi untuk mengonfigurasi proyek game Unreal Anda untuk plugin dan perbarui kode game Anda dengan blok kode yang disediakan untuk menambahkan fungsionalitas yang diperlukan untuk server dan build klien Anda.

- Unduhan plugin SDK
- Panduan: Integrasikan proyek Unity Anda dengan Amazon GameLift
- Panduan referensi: C # Server SDK 5 untuk Unity

Catatan: Plugin GameLift mandiri Amazon untuk Unity tidak mendukung penggunaan armada kontainer.

Untuk pengembang yang menggunakan mesin game lainnya

Ikuti panduan integrasi server dan klien umum ini:

- Integrasikan server game
- Integrasikan klien game

Amazon GameLift menawarkan pustaka SDK 5 server untuk bahasa berikut:

- Server SDK 5 untuk C ++ [Unduh SDK] [Panduan referensi]
- Server SDK 5 untuk C # [Unduh SDK] [Panduan referensi]
- Server SDK 5 untuk Go [Unduh SDK] [Panduan referensi]

Bangun server game Anda untuk Linux

Armada GameLift kontainer Amazon mendukung server game yang berjalan pada platform Linux. Berikut adalah beberapa tips untuk membangun server game Anda untuk target Linux:

- Jika Anda mengembangkan game Anda dengan mesin game Unity, editor game menyediakan dukungan bawaan tanpa persyaratan khusus untuk membangun Linux.
- Jika Anda mengembangkan game Anda di C ++, Anda harus menyertakan pustaka OpenSSL untuk Linux ketika Anda membangun server Amazon SDK GameLift untuk C++, dan ketika Anda membangun server game Anda. Sertakan juga pustaka yang sama di image container server game Anda.
- Jika Anda mengembangkan game Anda dengan Unreal Engine di Windows, pertimbangkan opsi ini:
  - Bekerja dengan Unreal Engine untuk menyiapkan rantai alat kompilasi silang.

- Siapkan ruang kerja Linux terpisah, atau gunakan alat seperti subsistem Windows untuk Linux (WSL). Anda dapat menggunakan lingkungan ini untuk menjalankan Unreal Editor di Linux untuk membangun server game Anda.

Uji integrasi Anda secara lokal

Anda dapat menguji integrasi game Anda secara lokal menggunakan GameLift Anywhere armada Amazon. Pendekatan ini adalah praktik terbaik untuk membantu mengisolasi masalah yang terkait langsung dengan integrasi. AnywhereArmada adalah alat yang berguna untuk menjalankan aplikasi uji dan skenario permainan, seperti memulai/menghentikan sesi permainan dan melacak koneksi pemain. Anda dapat membangun dan menguji secara iteratif lebih cepat dengan Anywhere armada, yang menawarkan visibilitas yang lebih besar ke dalam aktivitas hosting.

Lihat bantuan Uji integrasi Anda menggunakan GameLift Anywhere armada Amazon untuk menggunakan GameLift Anywhere armada Amazon untuk pengujian integrasi. Alur kerja untuk menyiapkan lingkungan pengujian terlihat seperti ini:

1. Siapkan perangkat lokal yang menjalankan Linux.
2. Siapkan Anywhere armada. Buat lokasi khusus untuk perangkat lokal Anda, buat Anywhere armada, lalu daftarkan perangkat lokal Anda sebagai komputasi di armada.
3. Dapatkan token otentikasi untuk server game Anda. Proses server terintegrasi Anda memerlukan token untuk mengautentikasi dengan GameLift layanan Amazon. Anda dapat menggunakan kembali token yang sama untuk beberapa proses server yang berjalan secara bersamaan Langkah ini hanya diperlukan saat menggunakan Anywhere armada untuk pengujian integrasi.

   > **Note**
   >
   > Token otentikasi bersifat sementara dan harus diperbarui secara teratur. Pertimbangkan untuk menambahkan skrip ke paket build server Anda untuk meminta token baru.

4. Perbarui kode server game Anda untukAnywhere. Saat berjalan di armada Anywhere, server game perlu memanggil tindakan SDK server `InitSdk()` (C++) (C#) (Unreal) dengan parameter server berikut. Langkah ini diperlukan hanya ketika menggunakan Anywhere armada untuk pengujian integrasi. Setelah Anda menambahkan GameLift Agen Amazon ke gambar kontainer Anda, ia menangani parameter ini secara otomatis.

Sebagai praktik terbaik, atur kode server Anda untuk menarik nilai-nilai ini dari variabel lingkungan atau dari argumen konsol yang Anda tentukan saat peluncuran.

- `webSocketUrl`— Gunakan nilai`GameLiftServiceSdkEndpoint`, yang dikembalikan dari panggilan ke`register-compute`.
- `processId`— Tetapkan pengenal unik untuk proses server.
- `fleetId`— Pengidentifikasi armada Anywhere, yang dikembalikan dari panggilan ke`create-fleet`.
- `authToken`— Token otentikasi yang valid, yang dikembalikan dari panggilan ke`get-compute-auth-token`.

5. Di komputer lokal Anda, siapkan perangkat lunak pembuatan server game Anda dan luncurkan proses server.

   Jika integrasi server Anda berhasil, proses server memanggil tindakan SDK server `InitSDK()` untuk membuat koneksi dengan GameLift layanan Amazon, diikuti dengan panggilan `ProcessReady()` untuk memberi tahu layanan bahwa layanan siap untuk meng-host sesi permainan.

6. Mulai sesi permainan. Jika Anda telah mengintegrasikan klien game Anda untuk meminta sesi permainan, Anda dapat menggunakannya untuk meminta sesi permainan baru. Jika tidak, gunakan perintah AWS CLI. create-game-session Amazon GameLift membuat `GameSession` objek dan memulai proses untuk memulai sesi permainan baru.

   Jika integrasi Anda berfungsi, Amazon GameLift memanggil proses server di workstation lokal Anda untuk memulai sesi permainan baru (menggunakan on`StartGameSession()` panggilan balik). Ketika sesi permainan siap untuk pemain, server memproses panggilan`ActivateGameSession()`. Sebagai tanggapan, Amazon GameLift memperbarui `GameSession` status dan informasi koneksi sehingga klien game dapat terhubung ke sesi permainan dan memainkan game.

Siapkan gambar kontainer dengan perangkat lunak server game Anda

Buat kumpulan gambar kontainer Linux yang Anda rencanakan untuk dijalankan di GameLift container fleet Amazon. Topik ini menjelaskan cara membuat gambar kontainer dengan perangkat lunak server game Anda dan GameLift Agen Amazon. Sebelum Anda memulai tugas ini, selesaikan

mengintegrasikan kode server game Anda dengan SDK GameLift server Amazon mengujinya dengan armada Amazon GameLift Anywhere (lihatIntegrasikan game Anda dengan Amazon GameLift).

> **Note**
>
> Amazon GameLift mendukung gambar kontainer Docker yang dibuat untuk berjalan di Linux. Seperti yang ditunjukkan dalam topik ini, Anda harus membangun gambar kontainer Anda di lingkungan Linux.

Topik

Siapkan direktori kerja Anda

Direktori kerja Anda adalah tempat Anda meletakkan semua file yang Anda butuhkan untuk membangun gambar kontainer Anda dan menentukan bagaimana Amazon GameLift menjalankannya.

Untuk menyiapkan direktori kerja kontainer

1. Buat direktori tempat Anda ingin bekerja dengan gambar GameLift wadah Amazon Anda.

   Example

   Sebagai contoh:

   ```
   [~/]$ mkdir -p work/glc/{target,gamebuild} && cd work && find .
   .
   ./glc
   ./glc/target
   ./glc/gamebuild
   ```

2. Kloning GameLift Agen Amazon.

Example

Sebagai contoh:

```
[~/work]$ git clone https://github.com/aws/amazon-gamelift-agent.git
Cloning into 'amazon-gamelift-agent'...
```

3. Membangun GameLiftAgent menggunakan Maven.

   Sebagai contoh:

   Example

   ```
   [~/work]$ cd amazon-gamelift-agent
   ```

   Example

   ```
   [~/work/amazon-gamelift-agent]$ mvn clean compile assembly:single && \
   mv target ../glc && cd ..
   ```

4. Tambahkan server game yang telah terintegrasi dengan server SDK 5.x, dibangun, dan dikemas ke dalam file. `.ZIP`

   - Salin `.ZIP` file Anda ke`~/work/glc/gamebuild/`.
   - Jika Anda hanya ingin belajar tentang armada kontainer, Anda dapat menggunakan contoh SimpleServer permainan kami.

   Example

   Unduh sampelnya.

   ```
   [~/work]$ curl -o glc/gamebuild/SimpleServer.zip 'https://ws-assets-prod-iad-r-iad-
   ed304a55c2ca1aee.s3.us-east-1.amazonaws.com/086bb355-4fdc-4e63-8ca7-af7cfc45d4f2/
   AmazonGameLiftSampleServerBinary.zip' && find glc
   <curl output>
   glc
   glc/target
   glc/target/GameLiftAgent-1.0.jar
   glc/gamebuild
   ```

```
glc/gamebuild/SimpleServer.zip
```

Bangun gambar kontainer Anda

Dockerfile Anda menentukan lingkungan, perangkat lunak, dan instruksi untuk membangun wadah Anda.

Untuk membuat Dockerfile

1. Pindah ke `glc` subdirektori.

   ```
   [~/work]$ cd $cd ~/work/glc
   ```

2. Buat dan buka Dockerfile baru.

   Example

   Sebagai contoh:

   ```
   [~/work/glc]$ vim Dockerfile
   ```

3. Salin konten dari template berikut dan tempel ke Dockerfile Anda.

Templat GameLift Dockerfile Amazon

```
# Base image
# ----------
  # Add the base image that you want to use over here,
  # Make sure to use an image with the same architecture as the
  # Instance type you are planning to use on your fleets.
  # We require JDK to be installed in the base image, so that
  # it can be used to run the &AGS; Agent
FROM public.ecr.aws/amazoncorretto/amazoncorretto:17-amd64
  #
# Game build directory
# --------------------
  # Add your game build to gamebuild directory and add the zip file name in the
 'GAME_BUILD_ZIP' env variable below.
  # The game build provided over here needs to be integrated with gamelift server sdk.
  # This template assumes that the game build is in a zip format.
```

```
ENV GAME_BUILD_ZIP="<ADD_GAME_BUILD_ZIP_FILE_NAME>" \
  #
# Default directory
# -----------------
  # Default directory, the value provided here should be where the game executable
 exists.
  # Provide this same value as your launch path in RuntimeConfiguration when creating a
 fleet.
  # Ref: https://docs.aws.amazon.com/gamelift/latest/apireference/
API_ServerProcess.html
GAME_EXECUTABLE="<ADD NAME OF EXECUTABLE WITHIN THE GAME BUILD>" \
HOME_DIR="/local/game" \
  #
  # Registered compute in anywhere fleet (not used in container fleets)
  # ---------------------------------------------------------------
  # Add the name for the registered compute in an anywhere fleet.
  # This environment variable is required only for anywhere fleets, but not for
 container fleets.
  # If it is set for container fleets, it will be overridden by Gamelift.
  GAMELIFT_COMPUTE_NAME="<ADD_COMPUTE_NAME>" \
  #
# Default Gamelift Agent jar
# --------------------------
GAMELIFT_AGENT_EXEC="GameLiftAgent-1.0.jar" \
  #
  # This env variable defines the name of the S3 bucket that stores the GameLift Agent
 logs.
  # This S3 bucket should exist in the customer AWS account.
  # In order to allow GameLift agent to upload logs to this s3 bucket, customers would
 need to
  # include s3:PutObject permission in the IAM role provided as instanceRoleArn during
 CreateFleet operation.
GAMELIFT_AGENT_LOGS_BUCKET_NAME="<ADD NAME OF GAMELIFT AGENT LOGS S3 BUCKET>" \
  #
# -----------
  # This env variable defines the name of the S3 bucket that stores the game session
 logs.
  # This S3 bucket should exist in the customer AWS account.
  # In order to allow GameLift agent to upload logs to this s3 bucket, customers would
 need to
  # include s3:PutObject permission in the IAM role provided as instanceRoleArn during
 CreateFleet operation.
# -----------
GAME_SESSION_LOGS_BUCKET_NAME="<ADD NAME OF GAME SESSION LOGS S3 BUCKET>" \
```

```
#
# -----------
GAMELIFT_AGENT_LOGS_PATH="/local/game/agentlogs/" \
  #
# NOT USED in container fleets - USED in Anywhere fleets
# ------------------------------------------------------------
  # Specifiy the type of compute resource used to host the game servers.
  # This env variable is required only for anywhere fleets, but not for container
 fleets.
  # If it is set for container fleets, it will be overridden by Gamelift.
#
# -----------
COMPUTE_TYPE="ANYWHERE" \
  #
  # Specify the credential to be used for creating the client.
  # This env variable is required only for anywhere fleets, but not for container
 fleets.
  # If it is set for container fleets, it will be overridden by Gamelift.
#
# -----------
CREDENTIAL_PROVIDER="environment-variable"

USER root

    # intall dependencies as necessary
    RUN yum install -y sudo \
                        unzip \
                        git \
                        shadow-utils \
                        iputils \
                        tar \
                        gcc \
                        make \
                        openssl-devel \
                        zlib-devel \
                        vim \
                        net-tools \
                        nc \
                        procps

    # Set up the ground for 'gamescale' user
    RUN groupadd -r gamescale -g 500 && \
      useradd -u 500 -r -g gamescale -m -s /sbin/nologin -c "Gamescale user" gamescale
 && \
```

```
      echo "gamescale ALL=(ALL) NOPASSWD: ALL" | (EDITOR="tee -a" visudo) && \
      mkdir -p $HOME_DIR && \
      mkdir $HOME_DIR/mono && \
      chown -R gamescale:gamescale $HOME_DIR

   WORKDIR $HOME_DIR

   # extract game build as necessary
   COPY ./gamebuild/$GAME_BUILD_ZIP .
   RUN unzip ./$GAME_BUILD_ZIP -d ./

   # copy Gamelift Agent jar
   COPY ./gameliftAgent/$GAMELIFT_AGENT_EXEC ./

   # Add permissions to game build and gamelift agent jar
   RUN chmod +x ./$GAME_EXECUTABLE
   RUN chmod +x ./$GAMELIFT_AGENT_EXEC

   # Check if java is installed on the image, if not then the Agent will not be able
to run
   RUN java --version

   USER gamescale

   ENV PATH "$PATH:$HOME_DIR/bin:$JAVA_HOME"

   # Change directory to bin
   WORKDIR $HOME_DIR

   # check path before starting the container
   RUN echo $PATH

   # Create logs directory for GameLift Agent & server processes
   RUN mkdir logs
   RUN mkdir agentlogs

   # Start the GameLift Agent
   ENTRYPOINT sleep 90 && java -jar $GAMELIFT_AGENT_EXEC -ip "192.168.1.1" -gslb
"$GAME_SESSION_LOGS_BUCKET_NAME" -galb "$GAMELIFT_AGENT_LOGS_BUCKET_NAME" -galp
"$GAMELIFT_AGENT_LOGS_PATH" -glc environment-variable
```

Template ini berisi instruksi minimum yang dibutuhkan wadah untuk dapat digunakan dalam GameLift armada Amazon.

Ubah konten sesuai kebutuhan untuk server game Anda.

> **Note**
>
> Catatan: Beberapa variabel lingkungan ini dapat diganti oleh. ContainerDefinition

1. Pindah ke `glc` subdirektori.

   ```
   [~/work]$ cd ~/work/glc/
   ```

2. Bangun gambar kontainer Anda. Anda dapat menentukan nama repositori lokal apa pun yang Anda inginkan.

   Example

   ```
   [~/work/glc]$ docker build -t <local repository name>:<optional tag> .
   ```

   Dalam contoh kami di bawah ini, kami menggunakan *`server sederhana`* sebagai nilai awal dan `version-1` sebagai `REPOSITORY` nilai. `TAG`

   Example

   ```
   [~/work/glc]$ docker build -t simple-server:version-1 .
   Successfully built 323c6829242d
   Successfully tagged simple-server:version-1
   ```

3. Lihat daftar gambar dan catat `REPOSITORY` dan `IMAGE ID` nilainya. Anda akan membutuhkannya dalam prosedur di bawah ini.

   Example

   ```
   [~/work/glc]$ docker images
       REPOSITORY       TAG          IMAGE ID        CREATED           SIZE
       simple-server    version-1    90c6661a02ed    14 minutes ago    1.24GB
   ```

Dorong gambar kontainer Anda ke Amazon ECR

Unggah gambar kontainer Anda ke repositori pribadi di Amazon ECR. Saat membuat definisi grup kontainer, Anda mereferensikan lokasi repositori ini sehingga Amazon GameLift dapat mengambil snapshot dari gambar penampung Anda dan menggunakannya saat menerapkan armada kontainer.

> **Note**
>
> Jika Anda belum memiliki repositori pribadi Amazon ECR, buatlah satu.

Dapatkan kredensyal Amazon ECR Anda

- Sebelum Anda dapat mendorong image container Anda ke Amazon ECR, Anda harus memperoleh AWS kredensialnya dalam bentuk sementara dan memberikannya ke Docker. Dapatkan kredensyal Amazon ECR Anda sehingga Docker dapat masuk.

  Example

  ```
  [~/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
  ```

Dorong gambar kontainer Anda ke Amazon ECR

1. Salin URI dari repositori pribadi Amazon ECR yang ingin Anda gunakan.
2. Terapkan tag Amazon ECR ke gambar kontainer Anda.

   Example

   ```
   [~/work/glc]$ docker tag <IMAGE ID from above> <Amazon ECR repository
    URI>:<REPOSITORY value from above>
   ```

3. Dorong gambar kontainer Anda ke Amazon ECR

Example

```
[~/work/glc]$ docker push <Amazon ECR repository URI>:<REPOSITORY value from above>
```

Pelajari selengkapnya di panduan pengguna Amazon ECR

- Membuat repositori pribadi
- Menggunakan Amazon ECR dengan AWS Command Line Interface
- Menandai repositori pribadi

Rancang armada GameLift kontainer Amazon

Topik-topik ini membahas keputusan penting yang akan Anda buat saat menyiapkan armada GameLift kontainer Amazon. Keputusan Anda memengaruhi cara Anda mengonfigurasi pengaturan untuk kontainer, grup kontainer, dan armada.

Topik

- Arsitek struktur kontainer armada Anda
- Tetapkan batas sumber daya
- Tentukan wadah penting
- Konfigurasikan koneksi jaringan untuk lalu lintas masuk
- Siapkan pemeriksaan kesehatan untuk wadah
- Tetapkan dependensi kontainer
- Konfigurasikan armada kontainer

Arsitek struktur kontainer armada Anda

Sebagai langkah pertama, identifikasi perangkat lunak dan sumber daya yang diperlukan untuk meng-host server game Anda, termasuk yang berikut:

- Aplikasi server game Anda. Aplikasi harus terintegrasi dengan GameLift fungsionalitas Amazon untuk hosting, termasuk server SDK versi 5+. Lihat Integrasikan game Anda dengan Amazon GameLift.

- GameLift Agen Amazon. Agen on-compute ini memelihara komunikasi dengan GameLift layanan Amazon dan mengelola siklus hidup semua proses server game. Untuk detail selengkapnya, lihat Server game dan GameLift Agen Amazon.
- Perangkat lunak dan sumber daya tambahan sesuai kebutuhan. Ini mungkin termasuk perangkat lunak yang diperlukan untuk menjalankan aplikasi server game Anda. Perangkat lunak pendukung umum digunakan untuk pencatatan dan pemantauan, keamanan, pengiriman konten, dan sinkronisasi data.

Selanjutnya, putuskan bagaimana menyusun perangkat lunak dan sumber daya Anda untuk armada GameLift kontainer Amazon. Amazon GameLift menggunakan grup kontainer untuk mengatur kontainer. Armada selalu memiliki satu grup kontainer replika dan secara opsional dapat memiliki armada kontainer daemon. Untuk detail selengkapnya, lihat Komponen armada kontainer.

- Mulailah dengan mendesain grup kontainer replika Anda. Pertimbangkan panduan berikut ini:
  - Bundel aplikasi server game Anda dan GameLift Agen Amazon ke dalam wadah yang sama. Jadikan wadah ini sebagai satu-satunya wadah penting grup replika.
  - Atur semua perangkat lunak lain untuk server game Anda ke dalam wadah. Anda dapat memilih untuk memasukkan semuanya ke dalam satu wadah di grup replika. Atau Anda dapat memilih untuk membuat satu atau lebih wadah sespan. Beberapa alasan untuk menggunakan sespan meliputi:
    - Untuk mengatur urutan startup/shutdown untuk perangkat lunak individual. Anda dapat mencapai ini dengan menempatkan perangkat lunak dalam wadah terpisah dan mengatur dependensi di antara mereka.
    - Untuk menetapkan batas khusus kontainer untuk penggunaan memori dan CPU.
    - Untuk menentukan pengaturan konfigurasi kontainer yang berbeda untuk setiap kontainer, seperti perintah peluncuran, titik masuk, direktori kerja, variabel lingkungan, atau pemeriksaan kesehatan.
- Putuskan apakah Anda memerlukan grup kontainer daemon untuk armada Anda. Pertimbangkan hal berikut:
  - Kontainer daemon biasanya digunakan untuk menjalankan proses latar belakang atau pemantauan.
  - Kontainer dalam grup daemon tidak direplikasi pada instance armada. Ini berarti bahwa kontainer dalam grup daemon tidak diskalakan bersama dengan grup kontainer replika.

- Grup daemon dapat memiliki beberapa kontainer. Anda dapat menetapkan wadah apa pun dalam grup daemon sebagai hal yang penting.

Tetapkan batas sumber daya

Untuk setiap grup kontainer, tentukan berapa banyak memori dan CPU yang dibutuhkan grup untuk menjalankan perangkat lunaknya. Amazon GameLift mengandalkan informasi ini untuk mengelola sumber daya untuk grup kontainer. Ini juga menggunakan informasi ini untuk menghitung berapa banyak grup kontainer replika yang dapat disimpan oleh gambar armada. Anda juga dapat menetapkan batas untuk wadah individu.

Tetapkan batas opsional untuk wadah

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.

- `SoftLimit`(memori) — Cadangan jumlah minimum memori untuk penggunaan eksklusif wadah. Wadah selalu memiliki jumlah cadangan yang tersedia untuk itu. Ini dapat melebihi minimum ini kapan saja, jika sumber daya tambahan tersedia.
- `HardLimit`(memori) - Tetapkan batas memori maksimum untuk wadah. Jika wadah melebihi batas ini, itu menghasilkan restart.
- `Cpulimit` — Cadangan jumlah minimum sumber daya CPU untuk penggunaan eksklusif container. 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

Beri tahu Amazon GameLift berapa banyak memori dan sumber daya CPU yang dibutuhkan setiap grup kontainer. Tujuannya adalah untuk mengalokasikan sumber daya yang cukup untuk mengoptimalkan kinerja server game. Amazon GameLift menggunakan batasan ini untuk menghitung cara mengemas grup kontainer replika pada instance armada. Anda juga akan menggunakannya saat memilih jenis instance untuk armada kontainer.

Hitung total memori dan CPU yang dibutuhkan untuk semua proses di setiap kontainer dalam grup. Pertimbangkan hal berikut:

- Proses apa yang berjalan di semua kontainer dalam grup kontainer? Tambahkan sumber daya yang diperlukan untuk proses ini.
- Berapa banyak proses server game bersamaan yang Anda rencanakan untuk dijalankan di setiap grup kontainer? Anda menetapkan nilai ini sebagai bagian dari konfigurasi runtime armada, tetapi Anda perlu merencanakan memori yang cukup untuk mereka di sini (lihat Optimalkan konfigurasi runtime Anda).

Berdasarkan perkiraan persyaratan grup kontainer Anda, tetapkan `ContainerGroupDefinition` properti berikut:

- `TotalMemoryLimit`— 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 jumlah semua batas memori lunak kontainer
  - sama dengan atau lebih besar dari batas memori keras tertinggi untuk wadah dalam grup
- `TotalCpuLimit` — Tetapkan batas CPU 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 kontainer. Sebagai praktik terbaik, pertimbangkan untuk mengatur nilai ini untuk menggandakan jumlah batas CPU kontainer.

Contoh skenario

Katakanlah kita mendefinisikan grup kontainer replika dengan tiga kontainer berikut:

- Wadah A adalah wadah replika penting kami. Ini menjalankan proses server game dan GameLift Agen Amazon. Kami memperkirakan kebutuhan sumber daya untuk satu server game di 512 MiB dan 1024 CPU. Kami berencana agar wadah menjalankan 10 proses server. Karena wadah ini menjalankan perangkat lunak kami yang paling penting, kami menetapkan cadangan memori lunak 6144 MiB dan tidak ada batas memori keras atau batas cadangan CPU.
- Container B menjalankan perangkat lunak pendukung dengan persyaratan sumber daya diperkirakan 1024 MiB dan 1536 CPU. Kami menetapkan batas cadangan memori lunak 1024 MiB, batas memori keras 2048 MiB, dan batas cadangan CPU 1024 CPU.
- Container C menjalankan pencatatan non-kritis dan utilitas pemantauan 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 (1) jumlah batas memori lunak (6144+1024 MiB), dan (2) batas memori keras tertinggi (1024 MiB).
- Total batas CPU: 13312 CPU. Nilai ini melebihi jumlah batas CPU (1024+512 CPU).

Tentukan wadah penting

Untuk setiap wadah, tentukan wadah sebagai penting atau tidak esensial. Semua grup kontainer harus memiliki setidaknya satu wadah penting. Wadah penting melakukan pekerjaan penting dari grup kontainer, seperti menghosting server game Anda. Wadah penting selalu diharapkan untuk berjalan. Jika gagal, seluruh grup kontainer akan dimulai ulang.

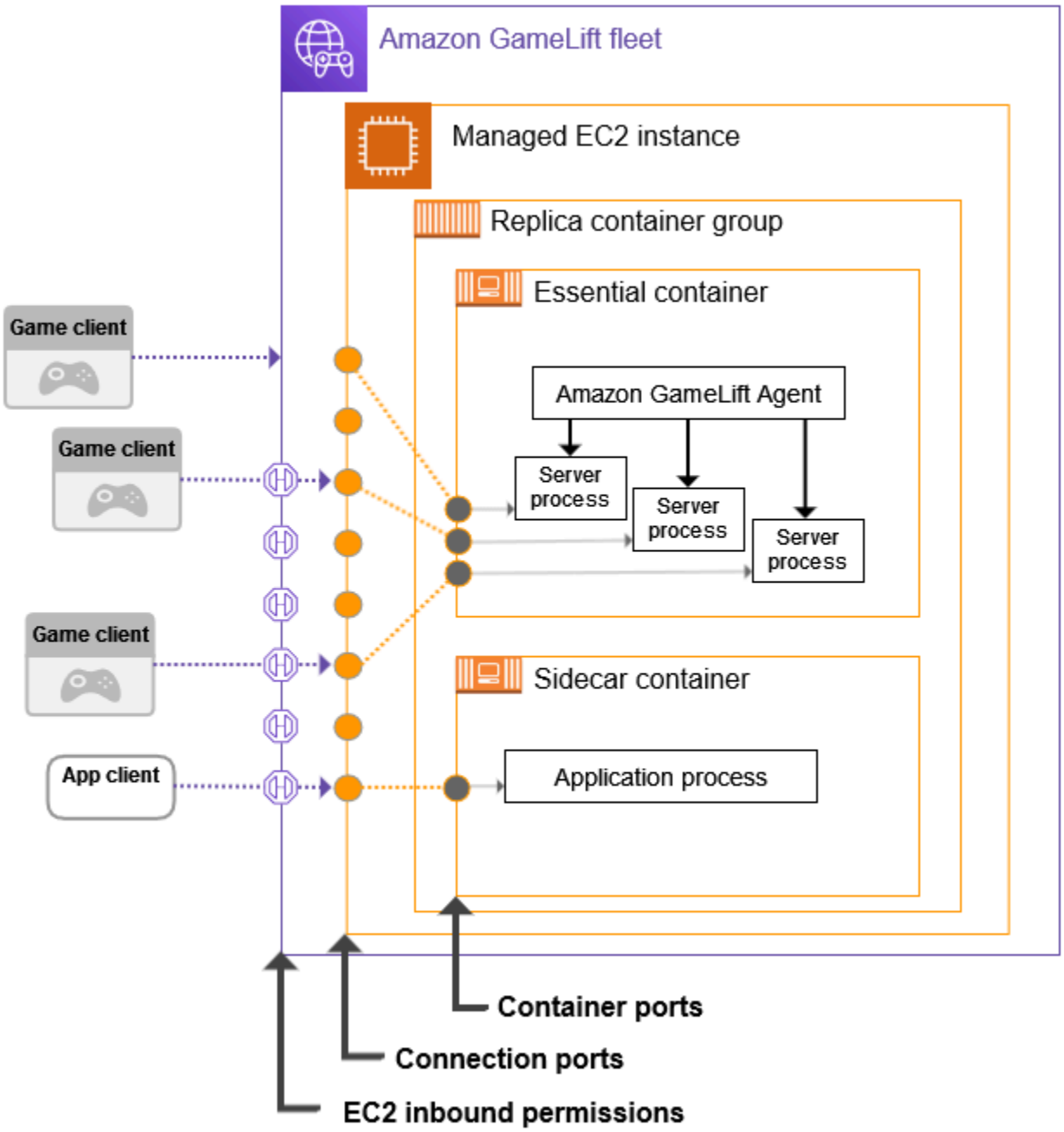
- Grup kontainer replika armada Anda dapat memiliki tepat satu kontainer penting. Wadah ini menjalankan GameLift Agen Amazon dan server game memprosesnya.
- Jika armada Anda memiliki grup kontainer daemon, Anda dapat menunjuk beberapa kontainer penting. Jadikan wadah daemon penting jika Anda ingin kegagalan kontainer untuk meminta restart grup kontainer.

Setel `ContainerDefinition` properti `Essential` ke true atau false untuk setiap kontainer.

Konfigurasikan koneksi jaringan untuk lalu lintas masuk

Anda dapat membuat akses jaringan yang memungkinkan lalu lintas eksternal terhubung ke kontainer apa pun dalam armada kontainer. Misalnya, wadah yang menjalankan proses server game Anda membutuhkan koneksi jaringan 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 Proses yang berjalan pada kontainer mendengarkan pada port kontainer, sementara lalu lintas masuk terhubung ke instance armada kontainer pada port koneksi. Amazon GameLift mempertahankan pemetaan antara port kontainer internal dan port koneksi yang menghadap eksternal, sehingga lalu lintas yang masuk diarahkan ke proses yang benar. Armada kontainer menggunakan izin masuk untuk mengelola akses ke koneksi. Anda tidak dapat mengubah konfigurasi port armada yang ada, tetapi Anda dapat menyesuaikan izin masuk untuk mengizinkan atau membatasi akses sesuai kebutuhan.



Atur rentang port kontainer

Konfigurasikan definisi kontainer Anda dengan port kontainer yang cukup untuk proses apa pun yang membutuhkan akses eksternal. Beberapa kontainer tidak membutuhkan port apa pun. Lainnya, seperti grup kontainer replika Anda, membutuhkan port yang cukup untuk menetapkan satu untuk setiap proses yang menjalankan server game. Proses server game mendengarkan port yang ditetapkan dan melaporkannya ke Amazon GameLift.

Saat Anda membuat definisi grup kontainer, tentukan rentang port kontainer untuk setiap kontainer (lihat ContainerDefinitionInput: PortConfiguration) yang membutuhkan akses jaringan. Pastikan jangkauannya cukup besar untuk menetapkan port ke setiap proses yang membutuhkannya. Proses harus diberi nomor port yang disertakan dalam konfigurasi port kontainer.

Atur rentang port koneksi

Konfigurasikan armada kontainer Anda dengan satu set port koneksi untuk lalu lintas masuk. Port koneksi menyediakan akses ke instance armada tempat kontainer Anda berjalan. Amazon GameLift menetapkan port koneksi dan memetakannya ke port kontainer sesuai kebutuhan.

Saat Anda membuat armada kontainer, tentukan rentang port koneksi (lihat ContainerGroupsConfiguration: ConnectionPortRange). Pastikan bahwa jangkauan memiliki port yang cukup untuk dipetakan ke setiap port kontainer di seluruh instance armada. Untuk menghitung port koneksi minimum yang diperlukan, gunakan rumus berikut:

```
[Total number of container ports defined for containers in the replica container group] * [Number of replica container groups per instance] + [Total number of container ports defined for containers in the daemon container group]
```

Sebagai praktik terbaik, gandakan jumlah minimum port koneksi.

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 properti armada ini untuk mengaktifkan dan mematikan akses jaringan sesuai kebutuhan, sehingga Anda hanya dapat membuka akses yang Anda butuhkan kapan saja.

Saat Anda membuat armada kontainer, tentukan satu set izin masuk (CreateFleetlihat:EC2). InboundPermissions Tetapkan properti port izin masuk untuk menyertakan beberapa atau semua nilai dalam pengaturan port koneksi armada. Untuk mengubah izin masuk pada armada kontainer yang ada, hubungi. UpdateFleetPortSettings

Contoh skenario

Katakanlah kita sedang mengkonfigurasi akses jaringan untuk armada kontainer.

- Grup kontainer replika armada kami memiliki satu kontainer, yang dirancang untuk menjalankan 10 proses server game bersamaan.

Saat membuat definisi grup kontainer replika, kami menetapkan `PortConfiguration` parameter untuk wadah ini sebagai berikut:

```
"PortConfiguration": {
  "ContainerPortRanges": [ { "FromPort": 10, "ToPort": 20, "Protocol": "TCP"} ]
  }
```

- Armada kami juga memiliki grup kontainer daemon. Ini memiliki 1 kontainer dengan 1 proses yang membutuhkan akses jaringan.

  Saat membuat definisi grup kontainer replika, kami menetapkan `PortConfiguration` parameter untuk wadah ini sebagai berikut:

  ```
  "PortConfiguration": {
    "ContainerPortRanges": [ { "FromPort": 25, "ToPort": 25, "Protocol": "TCP"} ]  }
  ```

- Armada kami dikonfigurasi untuk 3 salinan grup kontainer replika per instance armada. Dengan informasi ini, kami menggunakan rumus untuk menghitung jumlah port koneksi yang kami butuhkan:
  - Minimum: 31 port [10 port kontainer replika * 3 grup kontainer replika per instance+1 port kontainer daemon]
  - Praktik terbaik: 62 port [port minimum* 2]

  Saat membuat armada kontainer, kami mengatur `ConnectionPortRange` parameter `ContainerGroupsConfiguration` sebagai berikut:

  ```
  "ConnectionPortRange": { "FromPort": 1010, "ToPort": 1072 }
  ```

- Kami ingin mengizinkan akses ke semua port koneksi yang tersedia. Saat membuat armada kontainer, kami mengatur `EC2InboundPermissions` parameter sebagai berikut:

  ```
  "EC2InboundPermissions": [
    {"FromPort": 1010, "ToPort": 1072, "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 wadah penting, seluruh grup kontainer akan dimulai ulang.

Anda dapat menentukan kriteria khusus tambahan untuk mengukur kesehatan kontainer dan menggunakan pemeriksaan kesehatan untuk menguji kriteria tersebut. Untuk menyiapkan pemeriksaan kesehatan container, Anda dapat mendefinisikannya dalam image docker container atau dalam definisi container Anda. Jika Anda menetapkan pemeriksaan kesehatan dalam definisi kontainer, itu akan mengganti pengaturan apa pun dalam gambar kontainer.

Tetapkan pemeriksaan kesehatan opsional berdasarkan jenis wadah sebagai berikut:

- Untuk wadah replika penting, jangan mengonfigurasi pemeriksaan kesehatan. GameLiftAgen Amazon secara otomatis menangani pelaporan kesehatan untuk wadah ini.
- Untuk wadah replika yang tidak penting dan wadah daemon apa pun, Anda dapat mengatur parameter pemeriksaan kesehatan secara opsional.

Tetapkan `ContainerDefinition` 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 penampung 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 kontainer

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.

Konfigurasikan 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 menerapkan armada kontainer Anda ke Masing-masing Wilayah AWS yang GameLift didukung Amazon. 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, lihatLokasi GameLift hosting Amazon.

Pilih jenis dan ukuran instans untuk armada Anda

Amazon GameLift mendukung berbagai jenis instans Amazon EC2, 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 GameLift konsol Amazon (di bawah kuota Sumber Daya, Instance, dan 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 instans EC2. 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 wadah replika Anda dengan ruang terbuang 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 ketersediaan.

Optimalkan konfigurasi runtime Anda

Konfigurasi runtime armada adalah serangkaian instruksi tentang cara menjalankan proses server untuk hosting sesi game. Instruksi ini diterapkan oleh GameLift Agen Amazon di setiap grup kontainer replika di armada.

Konfigurasi runtime armada menentukan berapa banyak proses server yang berjalan secara bersamaan di setiap grup kontainer replika. Pengaturan ini memengaruhi cara Anda menghitung batas sumber daya grup kontainer dan cara Anda memilih jenis instans untuk armada Anda. Anda perlu menyeimbangkan ketiga elemen ini saat merancang armada Anda.

Untuk informasi selengkapnya tentang cara menggunakan konfigurasi runtime, lihat. Mengelola cara Amazon GameLift meluncurkan server game

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 sumber daya AWS lain dari armada.
- 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 GameLift kontainer Amazon

Definisi grup kontainer menjelaskan cara menerapkan aplikasi server game kontainer Anda ke armada kontainer. Ini adalah cetak biru yang mengidentifikasi set kontainer untuk dijalankan di armada dan cara menjalankannya. Saat membuat armada kontainer, Anda menentukan definisi grup kontainer yang akan diterapkan ke armada. Untuk informasi selengkapnya tentang grup kontainer, lihatKomponen armada kontainer.

Sebelum Anda mulai

Lakukan hal-hal berikut:

- Rancang arsitektur kontainer untuk menghosting server game Anda. Lihat Rancang armada GameLift kontainer Amazon.
- Rencanakan definisi kontainer untuk disertakan dalam grup kontainer. Jika Anda menggunakan AWS CLI, buat definisi wadah Anda dalam file JSON.
- Dorong gambar kontainer terakhir ke registri Amazon Elastic Container Registry (Amazon ECR) di tempat Wilayah AWS yang sama di mana Anda berencana untuk membuat grup kontainer. Amazon GameLift menyimpan snapshot dari setiap gambar pada saat Anda membuat definisi grup kontainer, dan menggunakan salinan saat menerapkan ke armada kontainer. Lihat Siapkan gambar kontainer dengan perangkat lunak server game Anda.
- Verifikasi bahwa AWS pengguna Anda memiliki izin IAM untuk mengakses repositori Amazon ECR. Minimal, Anda memerlukan izin untuk meneleponecr:DescribeImages.

Mengkloning definisi grup kontainer

Anda dapat menggunakan GameLift konsol Amazon untuk mengkloning definisi grup kontainer yang ada.

Untuk mengkloning grup kontainer

1. Di GameLift konsol Amazon, buka panel navigasi kiri dan pilih Grup kontainer.

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.

Buat definisi grup kontainer replika

Grup kontainer replika mengelola perangkat lunak server game Anda. Grup kontainer replika memiliki setidaknya satu kontainer yang menjalankan GameLift Agen Amazon dan proses server game Anda. Grup mungkin memiliki wadah “sespan” tambahan untuk menjalankan perangkat lunak pendukung.

Topik ini menjelaskan cara membuat definisi grup kontainer menggunakan GameLift konsol Amazon atau alat AWS CLI. Untuk informasi lebih rinci tentang pengaturan konfigurasi grup kontainer, lihat. Rancang armada GameLift kontainer Amazon

Console

Di GameLift konsol Amazon, pilih Wilayah AWS tempat Anda ingin membuat grup penampung.

Buka bilah navigasi kiri konsol dan pilih Grup kontainer. Pada halaman Container groups, pilih Create container group.

Langkah 1: Tentukan detail grup.

1. Masukkan nama definisi grup kontainer. Nama ini harus unik untuk Akun AWS dan Wilayah. Di konsol, definisi grup dicantumkan berdasarkan nama, sehingga dapat membantu untuk menetapkan label yang bermakna.
2. Pilih strategi penjadwalan replika.
3. Untuk batas memori Total, masukkan memori maksimum yang tersedia untuk grup kontainer. Untuk bantuan menghitung nilai ini, lihatTetapkan batas sumber daya.
4. Untuk batas CPU Total, masukkan daya komputasi maksimum yang tersedia untuk grup kontainer. Untuk bantuan menghitung nilai ini, lihatTetapkan batas sumber daya.

Langkah 2: Tambahkan definisi wadah.

Tentukan wadah dengan aplikasi server game Anda dan GameLift Agen Amazon. Ini adalah wadah replika penting Anda.

1. Berikan nama definisi kontainer. Setiap kontainer yang ditentukan untuk grup harus memiliki nilai nama yang unik.
2. Identifikasi URI image Amazon ECR dari image container. Masukkan 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. Untuk wadah Essential, Ya secara otomatis dipilih untuk definisi kontainer pertama. Jika Anda menambahkan definisi kontainer lain, Anda dapat mengaktifkan atau menonaktifkan pengaturan ini untuk setiap definisi. Untuk detail selengkapnya, lihat Tentukan wadah penting.
4. Atur satu atau lebih rentang port kontainer internal. Container ini meng-host server game Anda, jadi tentukan rentang dengan port yang cukup untuk setiap proses server untuk dijalankan dalam grup kontainer. Untuk detail selengkapnya, lihat Konfigurasikan koneksi jaringan untuk lalu lintas masuk.
5. Variabel Overrides dan Environment pengaturan opsional memungkinkan Anda menentukan nilai yang akan diteruskan ke wadah saat diluncurkan. Nilai yang Anda tetapkan di sini mengesampingkan pengaturan apa pun yang sudah ada dalam gambar kontainer.
6. Tetapkan Batas kontainer opsional untuk mengelola alokasi sumber daya untuk wadah ini. Untuk detail selengkapnya, lihat Tetapkan batas sumber daya.
7. Tentukan wadah non-esensial tambahan sesuai kebutuhan:

   - Berikan definisi wadah Nama dan URI gambar ECR. Kontainer yang tidak penting tidak boleh menjalankan GameLift Agen Amazon.
   - Tetapkan rentang port kontainer internal hanya jika kontainer memiliki proses yang membutuhkan akses jaringan.

- Secara opsional mengatur pemeriksaan Kesehatan untuk wadah. Ketika wadah yang tidak penting gagal dalam pemeriksaan kesehatan, itu meminta restart wadah yang gagal saja.
- Secara opsional mengatur Overrides, variabel Lingkungan, dan Batas alokasi sumber daya sesuai kebutuhan.

Langkah 3: Konfigurasikan dependensi.

Jika Anda memiliki lebih dari satu kontainer dalam definisi grup kontainer Anda, Anda dapat menentukan dependensi di antara mereka. Gunakan dependensi untuk mengatur urutan startup dan shutdown berdasarkan kondisi container. Untuk detail selengkapnya, lihat Tetapkan dependensi kontainer.

1. Identifikasi nama Container yang ingin Anda tambahkan dependensi. Wadah ini tidak dimulai sampai kondisi ketergantungan terpenuhi.
2. Identifikasi nama dan Kondisi Container dependensi. Wadah ini harus memenuhi kondisi sebelum wadah dependen dapat dimulai.
3. Tetapkan dependensi tambahan sesuai kebutuhan. Anda dapat membuat beberapa dependensi untuk wadah apa pun. Hindari membuat dependensi melingkar.

Langkah 4: Tinjau dan buat.

1. Tinjau semua pengaturan definisi grup kontainer Anda. Anda tidak dapat mengubah konfigurasi definisi grup kontainer setelah dibuat. 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 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-definitiondi Referensi Perintah AWS CLI.

Example : Kelompok kontainer replika

Contoh ini menggambarkan permintaan untuk definisi grup kontainer replika. Struktur perintah untuk membuat definisi grup replika dan daemon pada dasarnya identik. Rincian spesifik untuk setiap jenis grup dijelaskan dalam definisi wadah individu.

Contoh 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 \
    --scheduling-strategy REPLICA \
    --total-memory-limit 4096 \
    --total-cpu-limit 1024 \
    --container-definitions file://SimpleServer.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. Sebagai contoh:

   ```
   [~/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 ContainerDefinitionInputdi Referensi Amazon GameLift API.
3. Simpan file secara lokal sehingga Anda dapat merujuknya dalam perintah AWS CLI.

Example : Definisi wadah replika penting

Contoh ini menjelaskan wadah penting untuk grup kontainer replika Anda. Wadah replika penting mencakup aplikasi server game Anda, GameLift Agen Amazon, 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": "SimpleServer",
    "ImageUri": "111122223333.dkr.ecr.us-east-1.amazonaws.com/gl-containers:complex-
server",
    "Essential": true,
    "Cpu": 256,
    "MemoryLimits": {
      "HardLimit": 128
  },
     "PortConfiguration": {
           "ContainerPortRanges": [
              {
                 "FromPort": 2000,
                 "Protocol": "TCP",
                 "ToPort": 2100
              }
           ]
      }
    }
]
```

Buat armada GameLift kontainer Amazon

Saat Anda membuat definisi grup penampung, gunakan GameLift konsol Amazon atau AWS Command Line Interface (AWS CLI) untuk membuat armada kontainer.

Setelah Anda membuat armada baru, status armada melewati beberapa tahap saat Amazon GameLift 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. Untuk bantuan terkait masalah pembuatan armada, lihat Debug masalah GameLift armada Amazon.

Console

Di GameLift konsol Amazon, 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 Armada. Pada halaman Armada, pilih Buat armada.

Langkah 1: Pilih jenis komputasi

- Pilih jenis komputasi Containers.

Langkah 2: Tentukan detail armada

1. Di bagian Detail Armada, masukkan nama dan deskripsi armada.
2. Di bagian Detail grup Kontainer, identifikasi grup kontainer yang akan disebarkan ke armada. Anda harus menambahkan grup kontainer replika. Anda dapat menambahkan grup kontainer daemon secara opsional. Setiap kelompok harus dalam status`READY`.
3. Atur jangkauan port Connection untuk armada. Untuk detail selengkapnya, lihat Konfigurasikan koneksi jaringan untuk lalu lintas masuk.
4. Secara opsional tentukan replika yang Diinginkan per instance untuk digunakan. Anda dapat menentukan nomor yang diinginkan atau Anda dapat membiarkan Amazon GameLift menghitung jumlah maksimum yang mungkin. Jika Anda menentukan nomor yang diinginkan yang lebih besar dari maksimum yang dihitung, pembuatan armada akan gagal. Anda tidak dapat mengubah pengaturan ini setelah armada dibuat. Untuk detail lebih lanjut tentang pengepakan grup kontainer replika, lihatKonsep inti.
5. (Opsional) Di bawah Rincian tambahan:

a. Untuk peran Instans, tentukan peran IAM yang mengotorisasi aplikasi di build game Anda untuk mengakses AWS sumber daya lain di akun Anda. Untuk informasi selengkapnya, lihat Berkomunikasi dengan sumber daya AWS lain dari armada. Untuk membuat armada dengan peran instans, akun Anda harus memiliki `PassRole` izin IAM. Untuk informasi selengkapnya, lihat Contoh izin IAM untuk Amazon GameLift.

b. Untuk grup Metrik, Masukkan nama grup metrik armada baru atau yang sudah ada. Anda dapat menggabungkan metrik untuk beberapa armada dengan menambahkannya ke grup metrik yang sama.

Langkah 3: 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, lihatKonfigurasikan armada kontainer. Ukuran jenis instance yang Anda pilih akan memengaruhi bagaimana grup kontainer replika dikemas ke setiap instance armada. Bergantung pada pilihan Anda, pertimbangkan untuk meninjau pengaturan Anda untuk replika yang diinginkan per instance.

Langkah 4: Konfigurasikan runtime

Konfigurasi runtime menentukan bagaimana proses server game dimulai dan dijalankan. Instruksi ini diteruskan ke GameLift Agen Amazon, yang mengimplementasikannya dengan cara yang sama di setiap grup wadah replika. Anda dapat memperbarui konfigurasi runtime armada dengan menelepon UpdateRuntimeConfiguration.

1. Untuk jalur Peluncuran, masukkan jalur ke game yang dapat dieksekusi.
2. (Opsional) Untuk parameter Peluncuran, masukkan informasi untuk diteruskan ke game Anda yang dapat dieksekusi sebagai satu set parameter baris perintah.
3. Tentukan jumlah proses Bersamaan untuk tetap berjalan di setiap grup kontainer replika. Tinjau GameLift kuota Amazon tentang jumlah proses server per instance. Batasan pada proses server konkuren per instans berlaku untuk total proses konkuren untuk semua konfigurasi. Jika Anda mengonfigurasi armada untuk melebihi batas, armada tidak dapat diaktifkan.
4. Tetapkan batas opsional pada aktivasi sesi Game bersamaan. Pengaturan ini memungkinkan Anda membatasi jumlah sumber daya yang dikonsumsi saat memulai sesi permainan baru. Aktivasi sesi permainan dapat memiliki dampak kinerja pada sesi permainan yang ada.
5. Atur pengaturan port EC2 untuk memungkinkan lalu lintas eksternal mendapatkan akses ke proses yang berjalan di armada. Tentukan beberapa atau semua nomor port koneksi yang ditentukan untuk armada. Anda tidak perlu mengatur port ini saat membuat armada, tetapi tanpa mereka tidak ada lalu lintas yang dapat terhubung ke server game Anda. Untuk memperbarui pengaturan port armada nanti, hubungi UpdateFleetPortSettings
6. Di bawah Pengaturan sumber daya sesi game, konfigurasikan fitur opsional berikut:

    a. Aktifkan atau nonaktifkan kebijakan perlindungan penskalaan Game. Dengan perlindungan aktif, Amazon GameLift tidak akan mematikan instance selama acara penurunan skala jika mereka menyelenggarakan sesi permainan aktif.

    b. Tetapkan batas pembuatan Sumber Daya maksimum untuk membatasi jumlah sesi permainan yang dapat dibuat oleh satu pemain selama rentang waktu tertentu.

Langkah 5: Konfigurasikan tag

- (Opsional) Tambahkan tag ke build dengan memasukkan pasangan Kunci dan Nilai. Pilih Berikutnya untuk melanjutkan ke tinjauan pembuatan armada.

Langkah 6: 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 Kelola GameLift armada Amazon Anda. Anda dapat memperbarui kapasitas armada setelah armada mencapai status AKTIF. Untuk informasi selengkapnya, lihat Penskalaan kapasitas GameLift hosting Amazon. 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`, karena Amazon GameLift 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-fleet` perintah. Untuk informasi selengkapnya tentang perintah ini, lihat `create-fleet`di Referensi AWS CLI Perintah.

Contoh `create-fleet` permintaan yang ditunjukkan di bawah ini membuat armada kontainer baru dengan karakteristik sebagai berikut:

- ContainerGroupsConfiguration Menentukan definisi grup kontainer replika tunggal:. `MegaFrogRaceServer.NA.v2` Tiga salinan grup replika akan dikerahkan ke setiap instance armada. Setiap instance memiliki 30 port koneksi yang tersedia untuk akses ke proses pada instance.
- Armada menggunakan c5.large instans On-Demand.
- Ini menyebarkan grup kontainer ke lokasi berikut:
  - us-west-2 (Wilayah asal)

  - ca-central-1 (lokasi terpencil)
- Setiap grup kontainer replika pada sebuah instance akan menjalankan 5 proses server game secara bersamaan, memungkinkan setiap instance untuk meng-host hingga 15 sesi game sekaligus.
- Di setiap grup kontainer replika, Amazon GameLift memungkinkan dua sesi permainan baru untuk diaktifkan secara bersamaan. Ini juga mengakhiri sesi permainan pengaktifan jika mereka tidak siap untuk menjadi tuan rumah pemain dalam waktu 300 detik.
- Semua sesi game yang di-host pada instans dalam armada ini mengaktifkan perlindungan sesi game.
- Pemain individu dapat membuat tiga sesi game baru dalam jangka waktu 15 menit.

```
aws gamelift create-fleet \
    --name SampleFleet123 \
    --description "The sample test fleet" \
    --compute-type "CONTAINER" \
    --container-groups-configuration
 "ContainerGroupDefinitionNames=['MegaFrogRaceServer.NA.v2'],
 DesiredReplicaContainerGroupPerInstance=3,
 ConnectionPortRange={FromPort=1010,ToPort=1040}" \
    --ec2-instance-type c5.large \
    --region us-west-2 \
    --locations "Location=ca-central-1" \
    --fleet-type ON_DEMAND \
    --runtime-configuration "GameSessionActivationTimeoutSeconds=300,
 MaxConcurrentGameSessionActivations=2, ServerProcesses=[{LaunchPath=/local/game/
MegaFrogRace/server.exe,ConcurrentExecutions=5}]" \
    --new-game-session-protection-policy "FullProtection" \
    --resource-creation-limit-policy "NewGameSessionsPerCreator=3,
 PolicyPeriodInMinutes=15" \
    --ec2-inbound-permissions
 "FromPort=1010,ToPort=1040,IpRange=0.0.0.0/0,Protocol=UDP" \
```

Jika permintaan create-fleet berhasil, Amazon GameLift mengembalikan sekumpulan atribut armada yang menyertakan pengaturan konfigurasi yang Anda minta dan ID armada baru. Amazon GameLift 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-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

Kelola armada GameLift kontainer Amazon Anda

Ketika Anda ingin mendapatkan informasi tentang armada kontainer Anda atau membuat perubahan, Anda dapat menggunakan tindakan berikut untuk mengelola armada kontainer Anda.

Lihat sumber daya

Berikut adalah beberapa cara Anda bisa mendapatkan informasi tentang sumber daya di armada kontainer Anda.

- DescribeCompute- Mengembalikan rincian tentang grup kontainer terdaftar sebagai komputasi.
- DescribeContainerGroupDefinition- Mengembalikan rincian tentang definisi grup kontainer. Sumber daya ini menjelaskan bagaimana grup dan kontainernya dikonfigurasi.

- DescribeFleetAttributes- Mendapat atribut armada, yang mencakup jangkauan port koneksi, dan atribut lainnya.
- DescribeFleetCapacity- Mendapat hitungan kelompok kontainer replika di armada dan status mereka.
- DescribeRuntimeConfiguration- Menjelaskan proses server yang berjalan di setiap grup kontainer replika.
- GetComputeAccess- Menyediakan akses jarak jauh ke instance hosting grup kontainer.
- GetComputeAuthToken- Meminta token otentikasi dari Amazon GameLift untuk sumber daya komputasi dalam armada kontainer.
- ListCompute- Daftar grup kontainer terdaftar sebagai komputasi.
- ListContainerGroupDefinitions- Daftar definisi grup kontainer.
- ListFleets- Daftar armada yang menggunakan grup kontainer tertentu.

Perbarui sumber daya

Berikut adalah beberapa cara Anda dapat membuat dan memodifikasi sumber daya armada kontainer.

- CreateContainerGroupDefinition- Membuat definisi grup kontainer.
- CreateFleet- Membuat armada kontainer saat `ComputeType` diatur ke`CONTAINER`.
- RegisterCompute- Register menghitung dengan armada kontainer.
- UpdateFleetAttributes- Memperbarui atribut armada yang bisa berubah, seperti opsi konfigurasi armada Anywhere.
- UpdateFleetCapacity- Memperbarui pengaturan kapasitas untuk armada EC2 terkelola atau armada kontainer.
- UpdateRuntimeConfiguration- Memperbarui konfigurasi runtime, yang menjelaskan proses server apa yang akan dijalankan di setiap grup kontainer replika yang terdaftar sebagai komputasi.

Hapus sumber daya

Berikut adalah beberapa cara Anda dapat menghapus sumber daya armada kontainer.

- DeleteContainerGroupDefinition- Menghapus definisi grup kontainer.
- DeleteFleet- Menghapus armada.

- DeregisterCompute- Menghapus sumber daya komputasi dari armada kontainer.

Menskalakan armada GameLift kontainer Amazon

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, Anda menskalakan kapasitas armada Anda dengan menambahkan atau menghapus instance armada.

Saat Anda membuat armada baru, Amazon GameLift menetapkan kapasitas armada yang diinginkan ke satu instans dan menyebarkan satu instance di wilayah asal armada. Untuk armada multi-lokasi, Amazon GameLift menyebarkan satu instance ke wilayah asal dan ke setiap lokasi terpencil. Setelah status armada tercapai`ACTIVE`, Anda dapat meningkatkan kapasitas yang diinginkan untuk meningkatkan, atau menurunkan kapasitas yang diinginkan untuk menurunkan skala.

Anda dapat menggunakan fitur GameLift penskalaan Amazon untuk mengubah kapasitas secara manual atau mengatur penskalaan otomatis berdasarkan permintaan pemain:

- Siapkan penskalaan otomatis dengan pelacakan target. Lihat Mengatur kapasitas GameLift armada Amazon secara manual.
- Ubah kapasitas armada Anda secara manual. Lihat Penskalaan otomatis berbasis target.

Saat menskalakan armada kontainer, Anda perlu memahami 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 processes per replica container group] * [# of
     replica container groups per instance
    ```

    - Untuk proses per grup kontainer replika, panggil DescribeRuntimeConfigurationdan hitung jumlah eksekusi bersamaan untuk proses server game.

    - Untuk grup kontainer replika per instance, panggil DescribeFleetAttributesuntuk mendapatkan grup replika yang diinginkan per nilai instance. Jika nilai ini tidak disetel, gunakan `MaxReplicaContainerGroupsPerInstance` nilainya sebagai gantinya.
- 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:

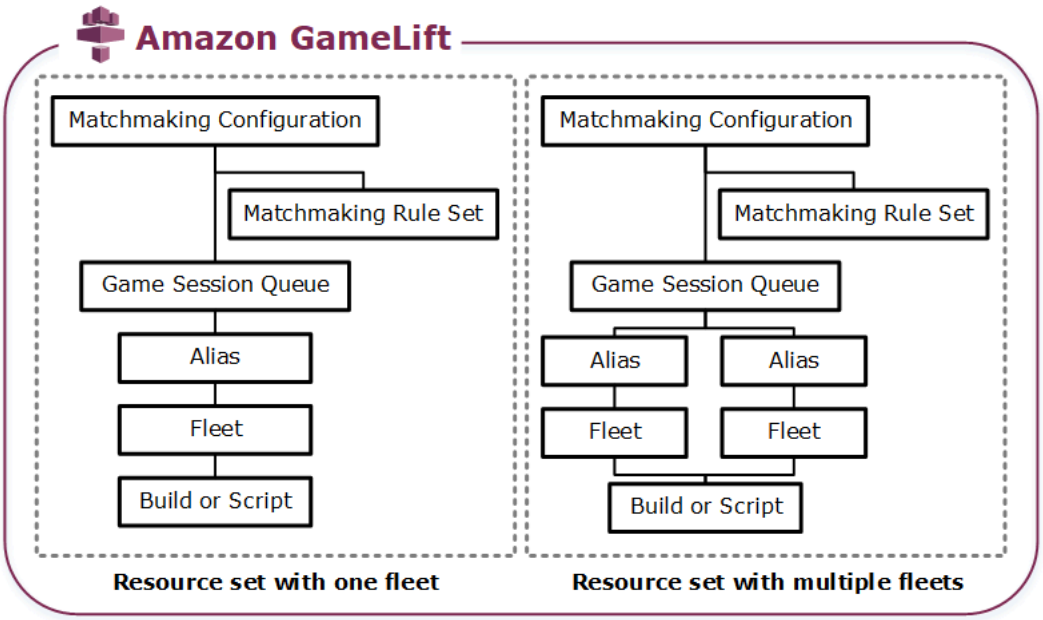
    ```
    [Players per instance] = [# of game sessions per instance] * [# of player slots per
     game session]
    ```

Untuk mendapatkan total kapasitas armada kontainer saat ini, hubungi DescribeFleetCapacityatau DescribeFleetLocation Kapasitas untuk mendapatkan jumlah grup kontainer replika dalam 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 replika.

Mengelola sumber daya GameLift hosting Amazon

Bagian ini memberikan informasi terperinci tentang menyiapkan sumber daya yang GameLift dikelola Amazon untuk menjalankan server game Anda dan menyelenggarakan sesi game untuk pemain. Anda harus mengonfigurasi dan menerapkan sumber daya, menskalakan kapasitas untuk memenuhi permintaan pemain, dan menemukan sumber daya yang tersedia untuk menyelenggarakan sesi game.

Diagram berikut menggambarkan bagaimana objek GameLift sumber daya Amazon berhubungan satu sama lain. Gunakan build atau script untuk membuat armada, memberikan armada alias, dan menambahkan armada ke antrian sesi permainan menggunakan alias mereka. Untuk game yang menggunakan FlexMatch perjodohan, gunakan antrian sesi permainan dan aturan perjodohan yang ditetapkan untuk membuat konfigurasi perjodohan.



Kode server game

- Build - Perangkat lunak server game yang dibuat khusus yang berjalan di Amazon GameLift dan menyelenggarakan sesi game untuk pemain Anda. Game build mewakili kumpulan file yang menjalankan server game Anda pada sistem operasi tertentu, dan yang harus Anda integrasikan dengan AmazonGameLift. Unggah file build game ke Amazon GameLift di Wilayah AWS tempat Anda berencana menyiapkan armada. Untuk informasi selengkapnya, lihat Unggah build server khusus ke Amazon GameLift.
- Skrip — Konfigurasi dan logika game kustom Anda untuk digunakan dengan Realtime Servers. Konfigurasikan Server Realtime untuk klien game Anda dengan membuat skrip

menggunakanJavaScript, dan tambahkan logika permainan khusus untuk menyelenggarakan sesi permainan untuk pemain Anda. Untuk informasi selengkapnya, lihat Unggah skrip Server Realtime ke Amazon GameLift.

Armada

Kumpulan sumber daya komputasi yang menjalankan server game Anda dan menyelenggarakan sesi game untuk pemain Anda. Untuk informasi tentang di mana Anda dapat menggunakan armada, lihat. Lokasi GameLift hosting Amazon Untuk informasi tentang membuat armada, lihatMenyiapkan GameLift armada Amazon.

Alias

Pengidentifikasi abstrak untuk armada yang dapat Anda gunakan untuk mengubah armada yang terhubung dengan pemain Anda kapan saja. Untuk informasi selengkapnya, lihat Tambahkan alias ke armada Amazon GameLift .

Antrian sesi permainan

Mekanisme penempatan sesi game yang menerima permintaan untuk sesi game baru dan mencari server game yang tersedia untuk menjadi tuan rumah sesi baru. Untuk informasi selengkapnya tentang antrean sesi game, lihat. Menyiapkan GameLift antrean Amazon untuk penempatan sesi game

Mengunggah build dan skrip ke Amazon GameLift

Sebelum menerapkan server game multipemain Anda untuk hosting dengan AmazonGameLift, Anda perlu mengunggah file server game Anda. Topik di bagian ini memberikan panduan tentang menyiapkan dan mengunggah file bangunan server game kustom atau file skrip server Realtime.

Topik

- Unggah build server khusus ke Amazon GameLift
- Unggah skrip Server Realtime ke Amazon GameLift

Unggah build server khusus ke Amazon GameLift

Setelah Anda mengintegrasikan server game Anda dengan Amazon GameLift, unggah file build ke Amazon GameLift. Topik ini mencakup cara mengemas file build game Anda, membuat skrip instalasi build opsional, lalu mengunggah file menggunakan AWS Command Line Interface(AWS CLI) atau AWS SDK.

Topik

Membuat paket file bangunan game Anda

Sebelum mengunggah server game yang dikonfigurasi ke Amazon GameLift, paketkan file build game ke dalam direktori build. Direktori ini harus mencakup semua komponen yang diperlukan untuk menjalankan server game dan sesi host game, termasuk yang berikut ini:

- 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, lihatDukungan pengembangan dengan Amazon GameLift.
- 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 GameLift server Amazon untuk C++ (termasuk yang dibuat dengan plugin Unreal), sertakan OpenSSL DLL untuk versi OpenSSL yang sama dengan yang Anda gunakan untuk membuat SDK server. Lihat file SDK README server untuk detail selengkapnya.

- Instal skrip (Opsional) - File skrip untuk menangani tugas yang menginstal build game Anda di server GameLift hosting Amazon. Tempatkan file ini di root direktori build. Amazon GameLift 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, lihatBerkomunikasi dengan sumber daya AWS lain dari armada.

Setelah Anda mengemas file build Anda, pastikan server game Anda dapat berjalan pada instalasi bersih OS target Anda. Ini memverifikasi bahwa Anda menyertakan semua dependensi yang diperlukan dalam paket Anda dan skrip penginstalan Anda akurat.

Buat GameLift build Amazon

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 membuat sumber daya build baru dengan ID build unik dan metadata lainnya. Build dimulai dalam status Inisialisasi. Setelah Amazon GameLift memperoleh file server game, build dipindahkan ke status Ready.

Saat build sudah siap, Anda dapat menerapkannya ke GameLift armada Amazon baru. Untuk informasi selengkapnya, lihat. Buat armada GameLift terkelola Amazon Saat Amazon GameLift menyiapkan armada baru, Amazon 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 CLIperintah. Perintah ini membuat catatan build baru di Amazon GameLift 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 Amazon GameLift server SDK 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 SDK GameLift server Amazon yang terintegrasi dengan server game Anda. Jika Anda tidak memberikan nilai, Amazon GameLift menggunakan nilai default`4.0.2`. Jika Anda menentukan versi SDK server yang salah, build server game mungkin gagal saat menelepon `InitSdk` untuk membuat sambungan ke GameLift layanan Amazon.
- 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" \
    --build-root "C:\mygame" \
    --name "My Game Nightly Build" \
    --build-version "build 255" \
    --region us-west-2
```

Menanggapi permintaan unggahan Anda, Amazon GameLift menyediakan kemajuan unggahan. Pada unggahan yang berhasil, Amazon GameLift 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 ke Amazon GameLift dari sana. Saat membuat build, Anda menentukan lokasi bucket S3, dan Amazon GameLift 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 Perhatikan label bucket dan nama file, Anda akan memerlukannya saat membuat GameLift build Amazon.
2. Berikan Amazon GameLift akses ke file build Anda. Buat peran IAM dengan mengikuti instruksi diMengakses file build game di Amazon S3. Setelah membuat peran, perhatikan nama sumber daya Amazon (ARN) peran baru, Anda akan memerlukannya saat membuat build.
3. Buat build. Gunakan GameLift konsol Amazon atau AWS CLI untuk membuat catatan build baru. Anda harus memiliki `PassRole` izin, seperti yang dijelaskan dalamContoh izin IAM untuk Amazon GameLift.

Console

1. Di GameLiftkonsol Amazon, 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 memberi Amazon GameLift akses ke bucket S3 dan objek build Anda.

4. (Opsional) Di bawah Tag, tambahkan tag ke build dengan memasukkan pasangan Kunci dan Nilai.
5. Pilih Create (Buat).

Amazon GameLift 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 fileAWS CLI.
2. Masukkan create-build perintah berikut:

```
aws gamelift create-build \
    --name user-defined name of build \
    --server-sdk-version Amazon GameLift server SDK 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 GameLift server Amazon yang Anda gunakan untuk mengintegrasikan server game Anda dengan Amazon GameLift. Jika Anda tidak memberikan nilai, Amazon GameLift 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 yang ditugaskan ke peran IAM yang Anda buat. Misalnya, “arn:aws:iam: :111122223333:role/”. GameLiftAccess Untuk melihat contoh kebijakan IAM, 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::111
gamelift" \
    --name "My Game Nightly Build" \
    --build-version "build 101" \
    --region us-west-2
```

3. Untuk melihat build baru, gunakan `describe-build`perintah.

Memperbarui file bangunan Anda

Anda dapat memperbarui metadata untuk sumber daya build menggunakan GameLift konsol Amazon atau perintah. `update-build`AWS CLI

Setelah membuat Amazon GameLift build, Anda tidak dapat memperbarui file build yang terkait dengannya. Untuk setiap set file baru, buat GameLift build Amazon baru. Menggunakan `upload-build`perintah, Amazon GameLift secara otomatis membuat catatan build baru untuk setiap permintaan. Jika Anda menyediakan file build menggunakan `create-build`perintah, unggah file.zip build baru dengan nama berbeda ke Amazon S3 dan buat build dengan mereferensikan nama file baru.

Coba tips berikut untuk men-deploy bangunan yang telah diperbarui:

- Gunakan antrian dan tukar armada sesuai kebutuhan. Saat menyiapkan klien game Anda dengan Amazon GameLift, tentukan antrian alih-alih armada. Dengan antrian, Anda dapat menambahkan armada baru dengan build baru ke antrian Anda dan menghapus armada lama. Untuk informasi selengkapnya, lihat Menyiapkan GameLift antrean Amazon untuk penempatan sesi game.
- Gunakan alias untuk mentransfer pemain ke build game baru. Saat mengintegrasikan klien game Anda dengan Amazon GameLift, tentukan alias armada, bukan ID armada. Untuk informasi selengkapnya, lihat Tambahkan alias ke armada Amazon GameLift .
- Siapkan pembaruan build otomatis. Untuk contoh skrip dan informasi tentang menggabungkan GameLift penerapan Amazon ke dalam sistem build Anda, lihat Mengotomatiskan Penerapan ke Amazon di Blog Teknologi Game. GameLift AWS

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 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 menggunakansudo. 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 mendukung bahasa interpreter 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 instance GameLift armada Amazon.

- Skrip penginstalan tidak dapat mengandalkan koneksi peering VPC. Koneksi peering VPC tidak tersedia sampai setelah Amazon GameLift menginstal instans build on fleet.

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.

```
vcredist_x64.exe /install /quiet /norestart /log c:\game\vcredist_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 berjalan dalam VPC layanan, Anda harus memberikan GameLift izin Amazon untuk mengambil peran AWS Identity and Access Management (IAM) atas nama Anda. GameLift Agar Amazon dapat mengambil peran, buat peran yang menyertakan kebijakan AWS terkelola`CloudWatchAgentAdminPolicy`, dan gunakan 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
```

Unggah skrip Server Realtime ke Amazon GameLift

Saat Anda siap untuk menerapkan Server Realtime untuk game Anda, unggah file skrip server Realtime yang telah selesai ke Amazon. GameLift Lakukan ini dengan membuat sumber daya GameLift skrip Amazon 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 baru, Amazon GameLift memberikan ID skrip unik (misalnya,`script-1111aaaa-22bb-33cc-44dd-5555eeee66ff`) dan mengunggah salinan file skrip. Waktu upload tergantung pada ukuran file skrip Anda dan kecepatan koneksi Anda.

Setelah Anda membuat sumber daya skrip, Amazon GameLift menerapkan skrip dengan armada Server Realtime baru. Amazon GameLift menginstal skrip server Anda ke setiap instans dalam armada, menempatkan file skrip. `/local/game`

Untuk memecahkan masalah aktivasi armada yang terkait dengan skrip server, lihat. Debug masalah GameLift armada Amazon

Buat paket file skrip

Skrip server Anda dapat menyertakan satu atau beberapa file yang digabungkan menjadi satu file.zip untuk diunggah. File.zip harus berisi semua file yang perlu dijalankan oleh 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 menyimpan file skrip secara lokal, Anda dapat mengunggahnya ke Amazon GameLift dari sana. Untuk membuat sumber daya skrip, gunakan GameLift konsol Amazon atau AWS Command Line Interface(AWS CLI).

Amazon GameLift console

Untuk membuat sumber daya skrip

1. Buka GameLiftkonsol Amazon.
2. Di panel navigasi, pilih Hosting, Skrip.
3. Pada halaman Skrip, pilih Buat skrip.
4. Pada halaman Create script, di bawah Script settings, 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 skrip, 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 Key dan Value.
6. Pilih Create (Buat).

   Amazon GameLift menetapkan ID ke skrip baru dan mengunggah file.zip yang ditunjuk. Anda dapat melihat skrip baru, termasuk statusnya, di halaman Skrip.

AWS CLI

Gunakan `create-script`AWS CLIperintah untuk menentukan script baru dan meng-upload file script 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 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 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 bucket Amazon S3 dan mengunggahnya ke Amazon GameLift dari sana. Saat membuat skrip, Anda menentukan lokasi bucket S3 dan Amazon GameLift mengambil file skrip Anda dari Amazon S3.

Untuk membuat sumber daya skrip

1. Simpan file skrip Anda dalam bucket S3. Buat file.zip yang berisi file skrip server Anda dan unggah ke bucket S3 dalam Akun AWS yang Anda kontrol. Perhatikan objek URI—Anda memerlukan ini saat membuat skrip AmazonGameLift.

   > **Note**
   >
   > Amazon GameLift tidak mendukung pengunggahan dari bucket S3 dengan nama yang berisi periode (.).

2. Berikan GameLift akses Amazon ke file skrip Anda. Untuk membuat peran AWS Identity and Access Management (IAM) yang memungkinkan Amazon GameLift mengakses bucket S3 yang berisi skrip server Anda, ikuti instruksi di dalamnya. Menyiapkan peran layanan IAM untuk Amazon GameLift Setelah Anda membuat peran baru, perhatikan namanya, yang Anda butuhkan saat membuat skrip.
3. Buat skrip. Gunakan GameLift konsol Amazon atau AWS CLI untuk membuat catatan skrip baru. Untuk membuat permintaan ini, Anda harus memiliki `PassRole` izin IAM, seperti yang dijelaskan dalamContoh izin IAM untuk Amazon GameLift.

Amazon GameLift console

1. Di GameLiftkonsol Amazon, di panel navigasi, pilih Hosting, Skrip.
2. Pada halaman Skrip, pilih Buat skrip.
3. Pada halaman Create script, di bawah Script settings, 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 URI Amazon S3.

d. Masukkan URI S3 objek skrip yang Anda unggah ke Amazon S3, lalu pilih versi Object. Jika Anda tidak mengingat 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 Key dan Value.

5. Pilih Create (Buat).

   Amazon GameLift menetapkan ID ke skrip baru dan mengunggah file.zip yang ditunjuk. Anda dapat melihat skrip baru, termasuk statusnya, di halaman Skrip.

AWS CLI

Gunakan `create-script`AWS CLIperintah untuk menentukan script baru dan meng-upload file script server Anda.

1. Buka jendela baris perintah dan beralih ke direktori di mana Anda dapat menggunakanAWS 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 mengembalikan objek skrip baru.

3. Untuk melihat skrip baru, hubungi `describe-script`.

Memperbarui file skrip

Anda dapat memperbarui metadata untuk sumber daya skrip menggunakan GameLift konsol Amazon atau perintah `update-script`AWS CLI.

Anda juga dapat memperbarui konten skrip untuk sumber daya skrip. Amazon GameLift menerapkan konten skrip ke semua instans armada yang menggunakan sumber daya skrip yang diperbarui. Saat skrip yang diperbarui diterapkan, instance menggunakannya saat memulai sesi game baru. Sesi game yang sudah berjalan pada saat pembaruan tidak menggunakan skrip yang diperbarui.

Untuk memperbarui file script

- Untuk file skrip yang disimpan secara lokal, untuk mengunggah file.zip skrip yang diperbarui, gunakan GameLift konsol Amazon atau perintah. update-script
- Untuk file skrip yang disimpan dalam bucket Amazon S3, unggah file skrip yang diperbarui ke bucket S3. Amazon GameLift secara berkala memeriksa file skrip yang diperbarui dan mengambilnya langsung dari bucket S3.

Menyiapkan GameLift armada Amazon

Bagian ini memberikan informasi terperinci tentang merancang, membangun, dan memelihara armada untuk digunakan dengan AmazonGameLift. Anda dapat menggunakan GameLift armada Amazon untuk menerapkan server game khusus dan Server Realtime.

Armada merepresentasikan sumber daya hosting Anda sebagai serangkaian instans Amazon Elastic Compute Cloud (Amazon EC2) atau perangkat keras fisik. Lokasi armada menentukan di mana instance atau perangkat keras digunakan untuk menyelenggarakan sesi game untuk pemain Anda. Ukuran armada, dan jumlah sesi permainan dan pemain yang dapat didukung, tergantung pada jumlah instance atau jumlah perangkat keras yang Anda berikan. Anda dapat menyesuaikan instance virtual secara manual atau dengan menggunakan penskalaan otomatis.

Banyak game dalam produksi yang menggunakan lebih dari satu armada. Anda dapat menggunakan beberapa armada, misalnya, untuk memiliki lebih dari satu versi server game Anda berjalan secara bersamaan, untuk menyediakan kapasitas cadangan untuk Armada Spot, atau untuk membangun redundansi.

Untuk mempelajari cara membuat armada yang dirancang untuk kebutuhan permainan Anda, mulailah denganPanduan desain GameLift armada Amazon. Setelah armada Anda berjalan, lihat, Penskalaan kapasitas GameLift hosting AmazonTambahkan alias ke armada Amazon GameLift , danMenyiapkan GameLift antrean Amazon untuk penempatan sesi game.

Topik

- Panduan desain GameLift armada Amazon
- Buat GameLift armada Amazon baru
- Kelola GameLift armada Amazon Anda
- Tambahkan alias ke armada Amazon GameLift
- Debug masalah GameLift armada Amazon
- Terhubung dari jarak jauh ke instance GameLift armada Amazon

Panduan desain GameLift armada Amazon

Panduan desain ini mencakup praktik terbaik untuk membuat armada sumber daya hosting untuk digunakan dengan AmazonGameLift. Pilih kombinasi sumber daya hosting dan pelajari cara mengonfigurasinya agar sesuai dengan permainan Anda.

Topik

- Memilih AmazonGameLiftmenghitung sumber daya
- Mengelola cara Amazon GameLift meluncurkan server game
- Menggunakan Instans Spot dengan Amazon GameLift

Memilih AmazonGameLiftmenghitung sumber daya

Untuk menyebarkan server game Anda dan menyelenggarakan sesi permainan untuk pemain Anda, AmazonGameLiftmenggunakanSumber daya Amazon Elastic Compute Cloud (Amazon EC2)disebutcontoh, atau perangkat keras fisik Anda. Saat menyiapkan armada baru menggunakan instance, tentukan jenis instance apa yang Anda butuhkan dan bagaimana menjalankan proses server game di dalamnya. Saat armada EC2 terkelola aktif dan siap menyelenggarakan sesi permainan, Anda dapat menambah atau menghapus instance sesuai kebutuhan untuk mengakomodasi permintaan pemain.

Anda dapat menggunakan Amazon AndaGameLiftserver game pada kombinasi dua jenis komputasi:

- EC2 yang dikelola— Armada EC2 terkelola menggunakan instans Amazon EC2 untuk meng-host server game Anda. AmazonGameLiftmengelola instance dan menghilangkan beban manajemen perangkat keras dan perangkat lunak dari hosting game Anda.
- AmazonGameLift Anywhere— AmazonGameLift Anywherearmada menggunakan infrastruktur yang ada untuk meng-host server game saat AmazonGameLiftmengelola perjodohan dan antrian Anda.

Saat Anda memilih sumber daya komputasi untuk armada Anda, pertimbangkan faktor-faktor berikut:

- Perangkat keras yang tersedia
- Lokasi armada
- Instans Sesuai Permintaan versus Instans Spot
- Sistem operasi
- Tipe instans
- Kuota layanan

Perangkat keras yang tersedia

Pertimbangkan infrastruktur yang ada dalam implementasi Anda. Saat Anda memigrasikan game ke AmazonGameLiftAnda dapat terus menggunakan infrastruktur Anda. Dengan AmazonGameLift Anywhere, Anda dapat menggunakan infrastruktur Anda sendiri bersama dengan AmazonGameLiftinstans EC2 terkelola. Anda juga dapat menggunakan infrastruktur yang ada untuk menghosting game yang lebih dekat dengan pemain Anda daripada Amazon yang didukungGameLiftlokasi dapat memungkinkan. Untuk informasi lebih lanjut tentang pengaturan AmazonGameLift Anywherearmada, lihatMembuat GameLift Anywhere armada Amazon.

Lokasi armada

Pertimbangkan lokasi geografis tempat Anda berencana untuk men-deploy server game Anda. Ketersediaan jenis instans bervariasi menurutWilayah AWSdan zona lokal.

Untuk armada multi-lokasi, ketersediaan instans dan kuota bergantung pada kombinasi Wilayah asal armada dan lokasi terpencil yang dipilih. Untuk informasi selengkapnya tentang lokasi armada, lihatLokasi GameLift hosting Amazon.

Untuk AmazonGameLift Anywherearmada, Anda menentukan lokasi perangkat keras fisik Anda. Untuk informasi selengkapnya tentang lokasi kustom, lihatAmazon GameLift Anywhere.

Instans Sesuai Permintaan versus Instans Spot

Instans On-Demand Amazon EC2 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 menggunakan yang tidak terpakaiAWSkapasitas komputasi. Harga Instans Spot berfluktuasi berdasarkan penawaran dan permintaan untuk setiap jenis instans di setiap lokasi.AWSdapat mengganggu Instans Spot kapan pun membutuhkan kapasitas kembali. AmazonGameLiftmenggunakan antrian dan algoritma FleetiQ untuk menentukan ituAWSakan mengganggu Instance Spot, itu menempatkan instance dalam keadaan daur ulang. Kemudian, ketika tidak ada sesi permainan aktif pada instance, AmazonGameLiftmencoba untuk menggantikannya.

Untuk informasi selengkapnya tentang cara menggunakan Instans Spot, lihatMenggunakan Instans Spot dengan Amazon GameLift.

Sistem operasi

AmazonGameLiftinstance mendukung build server game yang berjalan di Microsoft Windows atau Amazon Linux. Saat Anda mengunggah game build ke AmazonGameLift, tentukan sistem operasi untuk game. Saat Anda membuat armada Amazon EC2 untuk menerapkan build game, AmazonGameLiftsecara otomatis menyiapkan instance dengan sistem operasi build. Untuk informasi selengkapnya tentang sistem operasi server game yang didukung, lihatDukungan pengembangan dengan Amazon GameLift.

Saat menggunakan AmazonGameLift Anywherearmada, Anda dapat menggunakan sistem operasi apa pun yang didukung perangkat keras Anda. AmazonGameLift Anywherearmada mengharuskan Anda untuk menyebarkan build game Anda ke perangkat keras saat menggunakan AmazonGameLiftuntuk mengelola sumber daya Anda di satu tempat.

Tipe instans

Jenis instans armada Amazon EC2 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 (AWSGraviton).

> **Note**
>
> Instans Graviton Arm membutuhkan AmazonGameLiftserver 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. Contoh ini tidak memberikanout-of-the-boxdukungan 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 lebih lanjut:

- Tentang jenis instans, lihatJenis Instans Amazon EC2.
- Tentang menjalankan beberapa proses per instance, lihatMengelola cara Amazon GameLift meluncurkan server game.

Kuota layanan

Untuk melihat kuota layanan default untuk AmazonGameLift, dan kuota saat ini untuk AndaAkun AWS, lakukan hal berikut:

- Untuk informasi kuota layanan umum untuk AmazonGameLift, lihatAmazonGameLifttitik akhir dan kuotadiReferensi Umum AWS.

- Untuk daftar jenis instans yang tersedia per lokasi untuk akun Anda, bukaKuota layananhalaman AmazonGameLiftkonsol. 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, jalankanAWS Command Line Interface(AWS CLI) perintah`describe-ec2-instance-limits`. 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 diAmazonGameLiftkonsol. AmazonGameLifttim menggunakan kuesioner peluncuran untuk menentukan kuota dan batas yang benar untuk permainan Anda.

Mengelola cara Amazon GameLift meluncurkan server game

Anda dapat mengatur konfigurasi runtime armada EC2 yang dikelola untuk menjalankan beberapa proses server game per instans. Ini menggunakan sumber daya hosting Anda dengan lebih efisien.

Bagaimana armada mengelola beberapa proses

Amazon GameLift menggunakan konfigurasi runtime armada untuk menentukan jenis dan jumlah proses yang akan dijalankan pada setiap instans. 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 didefinisikan dalam konfigurasi runtime. Dengan beberapa proses, Amazon membuat peluncuran setiap proses GameLift terhuyung-huyung. Proses server memiliki rentang hidup yang terbatas. Saat berakhir, Amazon GameLift meluncurkan proses baru untuk mempertahankan jumlah dan jenis proses server yang ditentukan dalam konfigurasi waktu proses.

Anda dapat mengubah konfigurasi waktu aktif kapan saja dengan menambahkan, mengubah, atau menghapus konfigurasi proses server. Setiap instans secara teratur memeriksa pembaruan

konfigurasi runtime armada untuk mengimplementasikan perubahan. Berikut adalah cara Amazon GameLift mengadopsi perubahan konfigurasi runtime:

1. Instance mengirimkan permintaan ke Amazon GameLift untuk versi terbaru konfigurasi runtime.
2. Instance membandingkan proses aktifnya dengan konfigurasi runtime terbaru, dan kemudian melakukan hal berikut:
   - Jika konfigurasi runtime diperbarui menghapus jenis proses server, maka proses server aktif jenis ini terus berjalan sampai mereka berakhir. Instance tidak menggantikan proses server ini.
   - Jika konfigurasi runtime yang diperbarui mengurangi jumlah proses bersamaan untuk jenis proses server, maka proses server berlebih dari jenis ini terus berjalan hingga berakhir. Instance tidak menggantikan proses server berlebih ini.
   - Jika konfigurasi waktu proses yang diperbarui menambahkan jenis proses server baru atau meningkatkan proses bersamaan untuk jenis yang ada, maka instans memulai proses server baru, hingga GameLift maksimum Amazon. 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 Semua server game dalam build harus berjalan di platform yang sama dan menggunakan Amazon GameLift Server SDK.
- Buat konfigurasi waktu aktif dengan satu atau lebih konfigurasi proses server dan beberapa proses konkuren.
- Integrasikan klien game dengan AWS SDK versi 2016-08-04 atau yang lebih baru.

Untuk mengoptimalkan kinerja armada, kami menyarankan Anda melakukan hal berikut:

- Menangani skenario shutdown proses server sehingga Amazon GameLift dapat mendaur ulang proses secara efisien. Misalnya:
  - Tambahkan prosedur shutdown ke kode server game Anda yang memanggil API server`ProcessEnding()`.
  - Menerapkan fungsi callback `OnProcessTerminate()` dalam kode server game Anda untuk menangani permintaan penghentian dari Amazon. GameLift

- Pastikan Amazon GameLift mematikan dan meluncurkan kembali proses server yang tidak sehat. Laporkan status kesehatan kembali ke Amazon GameLift dengan menerapkan fungsi `OnHealthCheck()` callback dalam kode server game Anda. Amazon GameLift secara otomatis menutup proses server yang dilaporkan tidak sehat selama tiga laporan berturut-turut. Jika Anda tidak mengimplementasikan`OnHealthCheck()`, maka Amazon GameLift mengasumsikan bahwa proses server sehat, kecuali prosesnya gagal merespons komunikasi.

Pilih jumlah proses per instans

Saat memutuskan jumlah proses bersamaan yang akan dijalankan pada sebuah instance, ingatlah hal berikut:

- Amazon GameLift membatasi setiap instans 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 instans Amazon EC2 mungkin membatasi jumlah proses yang dapat dijalankan secara bersamaan. Uji konfigurasi yang berbeda untuk game Anda untuk menemukan jumlah proses yang tepat untuk jenis instans pilihan Anda.
- Amazon GameLift 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.

Menggunakan Instans Spot dengan Amazon GameLift

Saat menyiapkan armada EC2 GameLift terkelola Amazon Anda, Anda dapat menggunakan Instans Spot, Instans Sesuai Permintaan, atau kombinasi. Pelajari lebih lanjut tentang cara Amazon GameLift menggunakan Instans Spot diInstans Sesuai Permintaan versus Instans Spot. Untuk menggunakan armada spot, integrasi game Anda memerlukan penyesuaian yang tercantum di halaman ini.

Apakah Anda menggunakan FlexMatch untuk perjodohan? Anda dapat menambahkan Spot armada ke antrean sesi game yang ada untuk penempatan matchmaking.

1. Rancang antrean sesi game Anda untuk instans Spot.

   Mengelola penempatan sesi game dengan antrian adalah praktik terbaik, dan diperlukan saat menggunakan Instans Spot. Untuk mendesain antrian Anda, pertimbangkan hal berikut:

   - Lokasi - Untuk mencapai pengalaman pemain terbaik, pilih lokasi yang secara geografis dekat dengan pemain Anda.

- Jenis instans — Pertimbangkan persyaratan perangkat keras server game Anda dan ketersediaan instans di lokasi yang Anda pilih.

Untuk mencoba antrean yang mengoptimalkan ketersediaan dan ketahanan Spot, lihat. Tutorial: Mengatur antrean sesi game untuk Instans Spot

2. Buat armada untuk antrean yang dioptimalkan di SPOT Anda.

   Berdasarkan desain antrean Anda, buat armada untuk men-deploy server game Anda ke lokasi yang Anda inginkan dan tipe instans. Lihat Buat armada GameLift terkelola Amazon untuk membantu membuat dan mengonfigurasi armada baru.

3. Buat antrean sesi permainan Anda.

   Tambahkan tujuan armada, konfigurasikan 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 antrean.

   Saat klien game Anda menggunakan antrean untuk meminta sumber daya, antrean menghindari sumber daya dengan kemungkinan gangguan yang tinggi dan memilih lokasi yang sesuai dengan prioritas yang Anda tentukan. Untuk bantuan men-deploy penempatan sesi game di klien game Anda, lihatBuat sesi permainan.

5. Perbarui server game Anda untuk menangani gangguan Spot.

   AWSdapat mengganggu Instans Spot dengan notifikasi 2 menit, ketika perlu kapasitas kembali. Siapkan server game Anda untuk menangani gangguan guna meminimalkan dampak pemain.

   Sebelum mengambil AWS kembali Instans Spot, Instans Spot akan mengirimkan notifikasi penghentian. Amazon GameLift meneruskan notifikasi ke semua proses server yang terpengaruh dengan menjalankan fungsi callback Amazon GameLift Server SDK. `onProcessTerminate()` Menerapkan 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**
   >
   > AWSmelakukan segala upaya untuk memberikan notifikasi sebelum mengambil kembali dan misalnya, tetapi ada kemungkinan bahwa AWS merebut kembali Instans Spot

sebelum peringatan tiba. Siapkan server game Anda untuk menangani interupsi yang tidak terduga.

6. Tinjau performa armada dan antrean Spot Anda.

   Lihat GameLift metrik Amazon di GameLift konsol Amazon atau dengan Amazon CloudWatch untuk meninjau kinerja. Untuk informasi selengkapnya tentang GameLift metrik Amazon, lihatPantau Amazon GameLift dengan Amazon CloudWatch. Metrik kunci meliputi:

   - Tingkat gangguan - Gunakan `InstanceInterruptions` dan `GameSessionInterruptions` metrik untuk melacak jumlah dan frekuensi interupsi terkait SPOT untuk instance dan sesi game. 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 pada instance, sesi game, dan sesi pemain. Penggunaan untuk armada On-Demand Anda dapat menjadi indikator bahwa antrian menghindari penempatan ke armada Spot Anda untuk menghindari gangguan.

Buat GameLift armada Amazon baru

Buat armada baru dan men-deploy bangunan server game kustom Anda atau Realtime Servers untuk hosting. Anda dapat menerapkan sumber daya build atau skrip game apa pun yang Anda unggah ke AmazonGameLift.

Topik

- Cara kerja pembuatan GameLift armada Amazon
- Buat armada GameLift terkelola Amazon
- Membuat GameLift Anywhere armada Amazon

Cara kerja pembuatan GameLift armada Amazon

Saat Anda membuat armada baru, Amazon GameLift memulai alur kerja yang membuat armada dengan satu instans Amazon Elastic Compute Cloud (Amazon EC2) di setiap lokasi armada. Saat Amazon GameLift menyelesaikan setiap langkah alur kerja, armada memancarkan peristiwa

dan Amazon GameLift memperbarui status armada. Anda dapat melacak semua peristiwa menggunakan GameLift konsol Amazon atau dengan memanggil operasi Amazon GameLift API DescribeFleetEvents. Anda juga dapat melacak status masing-masing lokasi menggunakan DescribeFleetLocationAttributes.

Alur kerja pembuatan armada EC2:

- Amazon GameLift menciptakan sumber daya armada di Wilayah asal armada dan di setiap lokasi terpencil yang ditentukan dalam armada.
- Amazon GameLift menetapkan kapasitas yang diinginkan ke satu instans.
- Amazon GameLift menetapkan status armada dan lokasi ke New.
- Amazon GameLift mulai menulis peristiwa ke log peristiwa armada.
- Amazon GameLift mengalokasikan sumber daya komputasi yang diminta untuk satu instans baru di setiap lokasi armada.
- Amazon GameLift mengunduh file server game ke setiap instans dan menetapkan status armada untuk Mengunduh.
- Amazon GameLift memvalidasi file server game yang diunduh pada setiap instans untuk memverifikasi bahwa tidak ada kesalahan yang terjadi selama pengunduhan. Amazon GameLift menetapkan status armada untuk Memvalidasi.
- Amazon GameLift membangun server game pada setiap instans dan menetapkan status armada ke Building.
- Amazon GameLift mulai meluncurkan proses server pada setiap instans, mengikuti instruksi dalam konfigurasi waktu proses armada. Jika Anda mengonfigurasi armada untuk menjalankan beberapa proses server bersamaan per instans, maka Amazon akan GameLift menghentikan proses peluncuran beberapa detik. Karena setiap proses online, ia melaporkan kesiapan kembali ke AmazonGameLift. Amazon GameLift menetapkan status armada ke Mengaktifkan.
- Amazon GameLift menetapkan status armada dan status lokasi ke Aktif sebagai proses server melaporkan kesiapan.

Amazon GameLift Anywhere fleet creation

- Amazon GameLift menciptakan sumber daya armada. Untuk Wilayah asal armada dan setiap lokasi khusus yang ditentukan dalam armada, Amazon GameLift menetapkan status armada dan lokasi ke Baru.
- Amazon GameLift mulai menulis peristiwa ke log peristiwa armada.

- Setelah satu proses server dalam armada memberi tahu Amazon GameLift bahwa sudah siap, Amazon GameLift menetapkan status armada dan status lokasi ke Active. Saat proses server di lokasi armada lain melaporkan kesiapan, Amazon GameLift menetapkan status setiap lokasi armada ke Active.

Untuk bantuan mengatasi masalah pembuatan armada, lihat. Debug masalah GameLift armada Amazon

Buat armada GameLift terkelola Amazon

Gunakan GameLift konsol Amazon atau AWS Command Line Interface (AWS CLI) untuk membuat armada terkelola.

Setelah Anda membuat armada EC2 terkelola baru, status armada melewati beberapa tahap saat Amazon GameLift menyebarkan armada dan menginstal dan memulai server game. Armada siap menjadi tuan rumah sesi permainan, setelah mencapai `ACTIVE` status. Untuk bantuan terkait masalah pembuatan armada, lihat Debug masalah GameLift armada Amazon.

Console

Untuk membuat armada EC2 terkelola

1. Di GameLift konsol Amazon, di panel navigasi, pilih Armada.
2. Pada halaman Armada, pilih Buat armada.
3. Pilih EC2 yang Dikelola.
4. Pada halaman detail Armada lakukan hal berikut:
    a. Untuk Nama, masukkan nama armada. Sebaiknya sertakan jenis armada (Spot atau Sesuai Permintaan) dalam nama armada Anda. Hal ini mempermudah proses identifikasi jenis armada saat melihat daftar armada.
    b. Untuk Deskripsi, berikan deskripsi singkat tentang armada.
    c. Untuk tipe Biner, pilih Build atau Script untuk menentukan jenis server game yang GameLift diterapkan Amazon ke armada ini.
    d. Pilih Script atau Build dari daftar dropdown skrip atau build yang diunggah.
5. (Opsional) Di bawah Rincian tambahan untuk hal-hal berikut:
    a. Untuk peran Instans, tentukan peran IAM yang mengotorisasi aplikasi di build game Anda untuk mengakses AWS sumber daya lain di akun Anda. Untuk informasi

selengkapnya, lihat Berkomunikasi dengan sumber daya AWS lain dari armada. Untuk membuat armada dengan peran instans, akun Anda harus memiliki `PassRole` izin IAM. Untuk informasi selengkapnya, lihat Contoh izin IAM untuk Amazon GameLift.

Jika Anda ingin mengotorisasi aplikasi yang bukan server yang dapat dieksekusi, seperti CloudWatch agen, aktifkan opsi kredensyal bersama.

Anda tidak dapat memperbarui pengaturan ini setelah pembuatan armada.

b. Untuk pembuatan Sertifikasi, pilih untuk memiliki Amazon GameLift Menghasilkan sertifikat TLS untuk armada. Anda dapat menggunakan sertifikat TLS armada agar klien game Anda mengautentikasi server game saat menghubungkan, dan mengenkripsi semua komunikasi klien/server. Untuk setiap instance dalam armada yang mendukung TLS, Amazon GameLift juga membuat entri DNS baru dengan sertifikat. Gunakan sumber daya ini untuk menyiapkan autentikasi dan enkripsi untuk game Anda.

c. Untuk grup Metrik, Masukkan nama grup metrik armada baru atau yang sudah ada. Anda dapat menggabungkan metrik untuk beberapa armada dengan menambahkannya ke grup metrik yang sama.

Anda tidak dapat memperbarui grup metrik setelah pembuatan armada.

6. Pilih Selanjutnya.
7. Pada halaman Pilih lokasi, pilih satu atau beberapa lokasi jarak jauh tambahan untuk menerapkan instance. Wilayah beranda dipilih secara otomatis berdasarkan Wilayah tempat Anda mengakses konsol. Jika Anda memilih lokasi tambahan, instans armada juga digunakan di lokasi tersebut.

> **Important**
>
> Untuk menggunakan Wilayah yang tidak diaktifkan secara default, aktifkan di wilayah AndaAkun 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.

8. Pilih Selanjutnya.
9. Pada halaman Tentukan detail instance, pilih

   a. Instans sesuai permintaan atau Spot untuk armada ini. Untuk informasi selengkapnya tentang jenis armada, lihatInstans Sesuai Permintaan versus Instans Spot.

   b. Dari menu arsitektur Filter pilih x64 atau Arm.

   > **Note**
   >
   > Instans Graviton Arm memerlukan GameLift server Amazon 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).

   Untuk informasi tentang arsitektur Amazon EC2 Arm, lihat Prosesor AWSGraviton dan jenis instans Amazon EC2.

   Untuk informasi tentang jenis instance yang didukung oleh Amazon GameLift, lihat `EC2InstanceType` nilai di bawah parameter permintaan CreateFleet ().

10. Pilih jenis Instans Amazon EC2 dari daftar. Untuk informasi selengkapnya tentang memilih jenis instans, lihatTipe instans. Setelah membuat armada, Anda tidak dapat mengubah jenis instance.
11. Pilih Selanjutnya.
12. Pada halaman Configure runtime, di bawah konfigurasi Runtime lakukan hal berikut:

    a. Untuk jalur Peluncuran, masukkan jalur ke game yang dapat dieksekusi di build atau skrip Anda. Pada instans Windows, 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. (Opsional) Untuk parameter Peluncuran, masukkan informasi untuk diteruskan ke game Anda yang dapat dieksekusi sebagai satu set parameter baris perintah. Contoh:**`+sv_port 33435 +start_lobby`**.

c. Untuk proses Konkuren, pilih jumlah proses server yang akan dijalankan secara bersamaan pada setiap instance dalam armada. Tinjau GameLift batas Amazon pada jumlah proses server bersamaan.

   Batasan pada proses server konkuren per instans berlaku untuk total proses konkuren untuk semua konfigurasi. Jika Anda mengonfigurasi armada untuk melebihi batas, armada tidak dapat diaktifkan.

13. Di bawah aktivasi sesi Game, berikan batasan untuk mengaktifkan sesi permainan baru pada instans di armada ini:

    a. Untuk aktivasi sesi permainan bersamaan Max, masukkan jumlah sesi permainan pada instance yang diaktifkan pada saat yang bersamaan. Batas ini berguna saat meluncurkan beberapa sesi game baru yang dapat berdampak pada performa sesi game lain yang berjalan di instans.

    b. Untuk batas waktu aktivasi baru, masukkan berapa lama menunggu sesi diaktifkan. Jika sesi game tidak beralih ke `ACTIVE` status sebelum batas waktu, Amazon GameLift menghentikan aktivasi sesi game.

14. (Opsional) Di bawah pengaturan port EC2, lakukan hal berikut:

    a. Pilih Tambahkan pengaturan port untuk menentukan izin akses untuk lalu lintas masuk yang terhubung ke proses server yang digunakan pada armada.

    b. Untuk Jenis, pilih TCP Kustom atau UDP Kustom.

    c. Untuk rentang Port, Masukkan rentang nomor port yang memungkinkan koneksi masuk. Rentang port harus menggunakan format`nnnnn[-nnnnn]`, dengan nilai antara 1026 dan 60000. Contoh: **`1500`** atau **`1500-20000`**.

    d. Untuk rentang alamat IP, Masukkan berbagai alamat IP. Gunakan notasi CIDR. Contoh: **`0.0.0.0/0`** (Contoh ini memungkinkan akses ke siapa pun yang mencoba untuk connect.)

15. (Opsional) Di bawah Pengaturan sumber daya sesi permainan lakukan hal berikut:

    a. Untuk kebijakan perlindungan penskalaan Game, Aktifkan atau nonaktifkan perlindungan penskalaan. Amazon GameLift tidak akan menghentikan instans dengan perlindungan selama acara penurunan skala jika mereka menyelenggarakan sesi game aktif.

b. Untuk batas pembuatan Sumber Daya, masukkan jumlah maksimum sesi permainan yang dapat dibuat pemain selama periode kebijakan.

16. Pilih Selanjutnya.
17. (Opsional) Tambahkan tag ke build dengan memasukkan pasangan Kunci dan Nilai. Pilih Berikutnya untuk melanjutkan ke tinjauan pembuatan armada.
18. Pilih Create (Buat). Amazon GameLift memberikan ID ke armada baru dan memulai proses aktivasi armada. Anda dapat melacak status armada baru di halaman Armada.

Anda dapat memperbarui metadata dan konfigurasi armada kapan saja, terlepas dari status armada. Untuk informasi selengkapnya, lihat Kelola GameLift armada Amazon Anda. Anda dapat memperbarui kapasitas armada setelah armada mencapai status AKTIF. Untuk informasi selengkapnya, lihat Penskalaan kapasitas GameLift hosting Amazon. Anda juga dapat menambahkan atau menghapus lokasi jarak jauh.

AWS CLI

Untuk membuat armada denganAWS CLI, buka jendela baris perintah dan gunakan `create-fleet` perintah. Untuk informasi selengkapnya tentang `create-fleet` perintah, lihat `create-fleet`di Referensi AWS CLI Perintah.

Contoh permintaan `create-fleet` yang ditunjukkan di bawah ini membuat armada baru dengan karakteristik sebagai berikut:

- Armada menggunakan c5.large On-Demand Instances dengan sistem operasi yang sesuai untuk build game yang dipilih.
- Ini menyebarkan build server game yang ditentukan, yang harus dalam status Siap ke lokasi berikut:
  - us-west-2 (Wilayah asal)
  - sa-east-1 (lokasi arak jauh)
- Pembuatan sertifikat TLS diaktifkan.
- Setiap instans dalam armada akan menjalankan sepuluh proses server game yang identik secara konkuren, memungkinkan setiap instans menjadi host hingga sepuluh sesi game secara konkuren.
- Pada setiap contoh, Amazon GameLift memungkinkan dua sesi permainan baru untuk diaktifkan pada saat yang sama. Ini juga mengakhiri sesi permainan pengaktifan jika mereka tidak siap untuk menjadi tuan rumah pemain dalam waktu 300 detik.

- Semua sesi game yang di-host pada instans dalam armada ini mengaktifkan perlindungan sesi game.
- Pemain individu dapat membuat tiga sesi game baru dalam jangka waktu 15 menit.
- Setiap sesi permainan yang dihosting di armada ini memiliki titik koneksi yang termasuk dalam alamat IP dan rentang port yang ditentukan.
- Amazon GameLift menambahkan metrik untuk armada ini ke grup `EMEAfleets` metrik, yang (dalam contoh ini) menggabungkan metrik untuk semua armada di Wilayah EMEA.

```
aws gamelift create-fleet \
    --name SampleFleet123 \
    --description "The sample test fleet" \
    --ec2-instance-type c5.large \
    --region us-west-2 \
    --locations "Location=sa-east-1" \
    --fleet-type ON_DEMAND \
    --build-id build-92f061ed-27c9-4a02-b1f4-6f85b2385620 \
    --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=33435,IpRange=0.0.0.0/0,Protocol=UDP"
 "FromPort=33235,ToPort=33235,IpRange=0.0.0.0/0,Protocol=UDP" \
    --metric-groups  "EMEAfleets"
```

Jika permintaan create-fleet berhasil, Amazon GameLift mengembalikan sekumpulan atribut armada yang menyertakan pengaturan konfigurasi yang Anda minta dan ID armada baru. Amazon GameLift 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

Membuat GameLift Anywhere armada Amazon

Gunakan Amazon GameLift untuk mengintegrasikan perangkat keras dari lingkungan Anda ke hosting GameLift game Amazon Anda. Amazon GameLift Anywhere mendaftarkan perangkat keras Anda dengan Amazon GameLift dalam Anywhere armada. Anda dapat mengintegrasikan Anywhere dan mengelola armada EC2 dalam antrian matchmaker dan sesi game untuk mengelola perjodohan dan penempatan game.

Untuk informasi selengkapnya tentang menguji server game Anda dengan Amazon GameLiftAnywhere, lihatUji integrasi Anda menggunakan GameLift Anywhere armada Amazon.

Untuk memulai, Dukungan pengembangan dengan Amazon GameLift versi 5 atau lebih besar dan tinjau konsep penting berikut untuk menggunakan GameLift Anywhere armada Amazon.

Lokasi khusus

GameLiftAnywhereArmada Amazon menggunakan lokasi khusus untuk mewakili lokasi fisik infrastruktur Anda.

Registrasi perangkat

Agar GameLift Anywhere armada Amazon dapat berkomunikasi dengan sumber daya komputasi Anda, pertama-tama daftarkan perangkat Anda. Anda dapat menyelesaikan pendaftaran perangkat dari Amazon GameLift AWS SDK menggunakan `RegisterCompute`operasi. Operasi ini menggunakan alamat IP perangkat untuk mengaitkannya dengan lokasi armada dan berkomunikasi dengan AmazonGameLift.

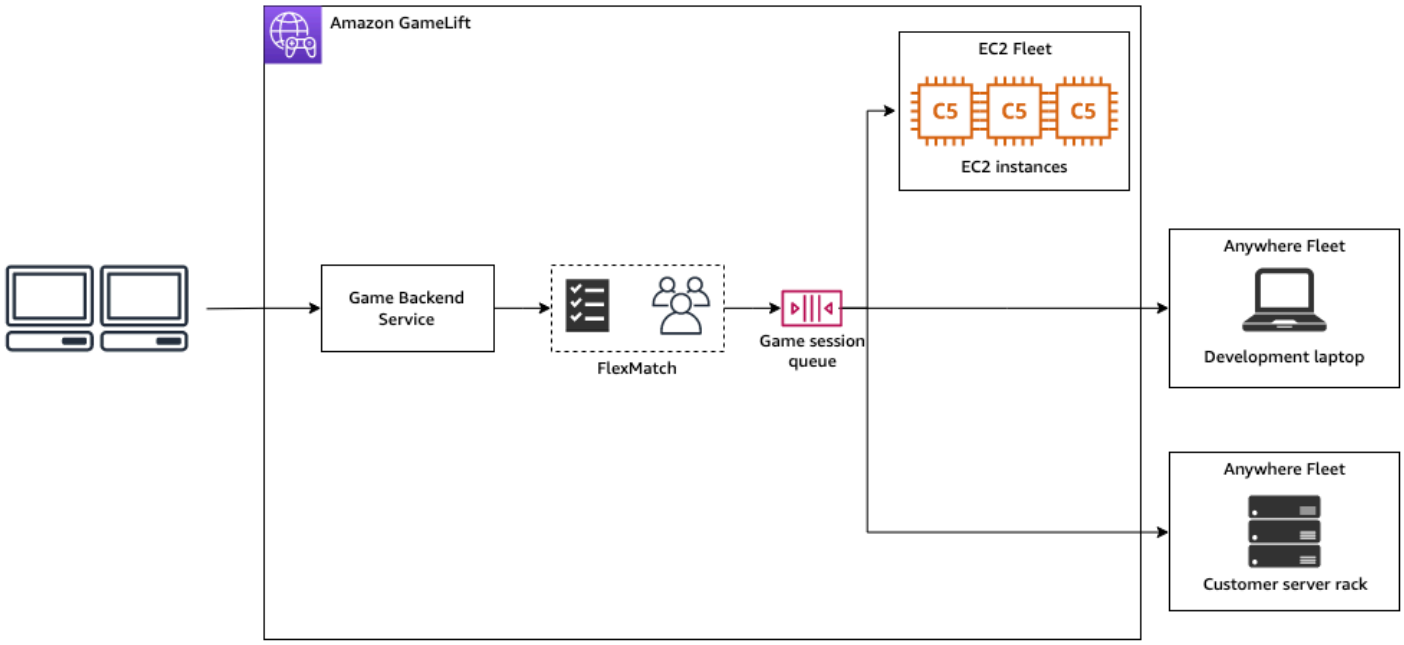
Token otentikasi

Saat Anda menginisialisasi server game di komputasi Anda, Amazon GameLift Server SDK menggunakan token autentikasi untuk mengautentikasi server game Anda ke Amazon. GameLift Anda dapat menggunakan kembali token autentikasi yang sama untuk semua server game pada komputasi yang sama, hingga waktu kedaluwarsa token autentikasi. Untuk mengambil token auth, panggil perintah `get-compute-auth-token`AWS Command Line Interface(AWS CLI). Berikan token ke setiap server game sesuai kebutuhan.

Sesi game

Setiap sesi permainan pada komputasi menggunakan token otentikasi yang sama yang dibuat saat mendaftarkan komputasi ke lokasi armada.

Diagram berikut menunjukkan antrian sesi permainan yang menggunakan FlexMatch perjodohan dan beberapa armada. Armada termasuk armada EC2 dengan instans C5, armada dengan laptop pengembangan, dan Anywhere armada dengan rak server yang Anywhere dihosting pelanggan.



Topik

- Membuat lokasi khusus
- Buat armada
- Daftarkan komputasi Anda
- Menjalankan proses server
- Buat sesi game
- Migrasi ke EC2 terkelola

Membuat lokasi khusus

Untuk memulai hosting game di sumber daya komputasi Anda, buat lokasi khusus yang menjelaskan tempat komputasi Anda berada.

Console

Untuk membuat lokasi kustom

1. Buka GameLiftkonsol Amazon.
2. Di panel navigasi, di bawah Hosting, pilih Lokasi.
3. Pada halaman Lokasi, pilih Buat lokasi.
4. Dalam Buat lokasi kotak dialog, lakukan hal berikut:

a. Masukkan nama Lokasi. Ini memberi label lokasi perangkat keras Anda yang GameLift digunakan Amazon untuk menjalankan game Anda dalam Anywhere armada. Amazon GameLift menambahkan nama lokasi kustom Anda dengan custom-.

b. (Opsional) Tambahkan tag sebagai pasangan kunci-nilai ke lokasi kustom Anda. Pilih Tambahkan tag baru untuk setiap tag yang ingin Anda tambahkan.

c. Pilih Create (Buat).

AWS CLI

Buat lokasi khusus menggunakan `create-location`perintah. `location-name`Label lokasi perangkat keras Anda yang GameLift digunakan Amazon untuk menjalankan game Anda dalam Anywhere armada. Saat membuat lokasi khusus Anda, nama lokasi harus dimulai dengan`custom-`.

```
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

Gunakan GameLiftkonsol Amazon atau AWS CLI untuk membuat Anywhere armada.

Setelah Anda membuat Anywhere armada baru, status armada bergerak dari `NEW` ke. `ACTIVE` Ketika mencapai `ACTIVE` status, armada siap untuk menjadi tuan rumah sesi permainan. Untuk bantuan terkait masalah pembuatan armada, lihat Debug masalah GameLift armada Amazon.

Console

Untuk membuat Anywhere armada

1. Buka GameLiftkonsol Amazon.
2. Di panel navigasi, di bawah Hosting, pilih Armada.
3. Pada halaman Armada, pilih Buat armada.
4. Pada langkah Compute type, pilih Anywhere, lalu pilih Next.
5. Pada langkah detail Armada, tentukan detailnya, lalu pilih Berikutnya.
6. Pada langkah Lokasi khusus, pilih lokasi khusus yang Anda buat, lalu pilih Berikutnya. Amazon GameLift secara otomatis memilih rumah Wilayah AWS sebagai Wilayah tempat Anda membuat armada. Anda dapat menggunakan Wilayah beranda untuk mengakses dan menggunakan sumber daya Anda.
7. Selesaikan langkah-langkah pembuatan armada yang tersisa, lalu pilih Kirim untuk membuat Anywhere armada Anda.

AWS CLI

Buat Anywhere armada menggunakan `create-fleet`perintah. Sertakan lokasi khusus Anda di`locations`. Amazon GameLift membuat armada di Wilayah asal Anda dan di lokasi khusus yang Anda berikan. Pada contoh berikut, ganti *`FleetName`* dan *`custom-location-1`* dengan informasi Anda sendiri. Variabel `custom-location-1` adalah nama lokasi yang dibuat dalam Membuat lokasi khusus langkah.

```
aws gamelift create-fleet \
--name FleetName \
--compute-type ANYWHERE \
--locations "Location=custom-location-1"
```

Contoh keluaran

```
{
    "FleetAttributes": {
        "FleetId": "fleet-cebb4da2-52a8-4c27-9b85-587f945c6445",
        "FleetArn": "arn:aws:gamelift:us-east-1:111122223333:fleet/fleet-
cebb4da2-52a8-4c27-9b85-587f945c6445",
        "Name": "HardwareAnywhere",
        "CreationTime": "2023-02-23T17:57:42.293000+00:00",
```

```
        "Status": "ACTIVE",
        "MetricGroups": [
            "default"
        ],
        "CertificateConfiguration": {
            "CertificateType": "DISABLED"
        },
        "ComputeType": "ANYWHERE"
    }
}
```

Daftarkan komputasi Anda

Untuk mendaftarkan sumber daya komputasi Anda di armada yang Anda buat, gunakan perintah `register-compute`. Ganti *`fleet-id`* dengan yang `fleet-id` dikembalikan pada langkah sebelumnya atau armada ARN yang ditemukan di halaman detail armada Anda di konsol. Ganti*`compute-name`*, dan *`ip-address`* dengan alamat IP sumber daya komputasi Anda.

> **Note**
>
> Amazon GameLift merekomendasikan untuk memanggil `get-compute-auth-token` perintah `register-compute` dan perintah dari skrip atau manajer proses yang terpisah dari server game Anda.

```
aws gamelift register-compute \
    --compute-name HardwareAnywhere \
    --fleet-id arn:aws:gamelift:us-east-1:111122223333:fleet/fleet-
cebb4da2-52a8-4c27-9b85-587f945c6445 \
    --ip-address 10.1.2.3 \
    --location custom-location-1
```

Contoh keluaran

```
{
    "Compute": {
        "FleetId": "fleet-cebb4da2-52a8-4c27-9b85-587f945c6445",
        "FleetArn": "arn:aws:gamelift:us-east-1:111122223333:fleet/fleet-
cebb4da2-52a8-4c27-9b85-587f945c6445",
```

```
        "ComputeName": "HardwareAnywhere",
        "ComputeArn": "arn:aws:gamelift:us-east-1:111122223333: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-east-1.api.amazongamelift.com"
    }
}
```

Menjalankan proses server

1. Dapatkan token autentikasi untuk sumber daya komputasi Anda dari armada yang Anda buat.

   Server game Anda menggunakan token autentikasi untuk mengautentikasi dengan AmazonGameLift. Setiap token otentikasi memiliki token kedaluwarsa. Untuk terus menggunakan sumber daya komputasi untuk meng-host server game Anda, ambil token autentikasi baru sebelum kedaluwarsa.

   > **Note**
   >
   > Amazon GameLift merekomendasikan untuk memanggil `get-compute-auth-token` perintah `register-compute` dan perintah dari skrip atau manajer proses yang terpisah dari server game Anda.

   Pada contoh berikut, ganti `fleet-id` dengan ARN atau ID armada armada yang dibuat pada langkah sebelumnya. Ganti `compute-name` dengan nama komputasi yang Anda buat menggunakan `register-compute` perintah pada langkah sebelumnya.

   ```
   aws gamelift get-compute-auth-token \
       --fleet-id arn:aws:gamelift:us-east-1:111122223333:fleet/fleet-
   cebb4da2-52a8-4c27-9b85-587f945c6445 \
       --compute-name HardwareAnywhere
   ```

   Contoh keluaran:

   ```
   {
       "FleetId": "fleet-cebb4da2-52a8-4c27-9b85-587f945c6445",
   ```

```
    "FleetArn": "arn:aws:gamelift:us-east-1:111122223333:fleet/fleet-
cebb4da2-52a8-4c27-9b85-587f945c6445",
    "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"
}
```

2. Jalankan instance server game Anda yang dapat dieksekusi.

   Untuk menjalankan server game Anda, inisialisasi server game Anda dengan menelepon `InitSDK()` dan meneruskannya parameter server Anda. Untuk informasi selengkapnya tentang parameter server, lihatServerParameters.

   Masukan SDK Server:

```
//Define the server parameters
ServerParameters serverParameters = new ServerParameters(
    webSocketUrl=wss://us-east-1.api.amazongamelift.com,
    processId=PID1234,
    hostId=HardwareAnywhere,
    fleetId=arn:aws:gamelift:us-east-1:111122223333:fleet/fleet-
cebb4da2-52a8-4c27-9b85-587f945c6445,
    authToken=0c728041-3e84-4aaa-b927-a0fb202684c0);

//InitSDK establishes a connection with GameLift's websocket server for
 communication.
var initSDKOutcome = GameLiftServerAPI.InitSDK(serverParameters);
```

3. Setelah proses server siap untuk menjadi tuan rumah sesi permainan, hubungi `ProcessReady()` dari server game Anda ke AmazonGameLift. Untuk informasi selengkapnya tentang parameter proses, lihat ProcessParameters

```
// Set parameters and call ProcessReady
var processParams = new ProcessParameters(
   this.OnStartGameSession,
   this.OnProcessTerminate,
   this.OnHealthCheck,
   this.OnUpdateGameSession,
   port=1024,
   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);
```

Buat sesi game

1. Tambahkan logika ke server game Anda sehingga proses server Anda merespons `onStartGameSession()` pesan dengan`ActivateGameSession()`. Operasi ini tidak memiliki parameter, tetapi mengirimkan pengakuan ke Amazon GameLift bahwa server Anda menerima dan menerima pesan sesi buat game.

   ```
   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();
   }
   ```

2. Dari layanan backend klien game Anda, mulai sesi permainan Anda menggunakan [start-matchmaking](), [start-game-session-placement](), atau [create-game-session]()perintah.

   ```
   aws gamelift create-game-session \
       --fleet-id arn:aws:gamelift:us-east-1:682428703967:fleet/fleet-
   cebb4da2-52a8-4c27-9b85-587f945c6445 \
       --name GameSession1 \
       --maximum-player-session-count 2 \
       --location custom-location-1
   ```

   Contoh keluaran:

   ```
   GameSession {
     FleetId = arn:aws:gamelift:us-east-1:682428703967:fleet/fleet-
   cebb4da2-52a8-4c27-9b85-587f945c6445,
     GameSessionId = 4444-4444,
     Name = GameSession1,
   ```

```
  Location = custom-location-1,
  IpAddress = 10.2.3.4,
  Port = 1024,
  ...
}
```

Amazon GameLift mengirimkan `onStartGameSession()` pesan ke proses server terdaftar Anda. Pesan berisi `GameSession` objek dari langkah sebelumnya dengan properti game, data sesi game, data mak comblang, dan lainnya tentang sesi permainan.

3. Saat sesi permainan selesai, akhiri proses server game.

   Masukan SDK Server:

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

4. Mulai proses server game lain dengan meneleponProcessReady(processParams).

Migrasi ke EC2 terkelola

Setelah Anda mengembangkan server game Anda dan Anda siap untuk mempersiapkan produksi, Anda dapat memiliki Amazon GameLift mengelola perangkat keras Anda. Untuk bermigrasi ke armada EC2 yang dikelola, unggah build Anda ke Amazon GameLift dan buat armada EC2 yang dikelola. Untuk informasi lebih lanjut tentang mengunggah build Anda dan menyiapkan armada, lihat Unggah build server khusus ke Amazon GameLift dan. Buat armada GameLift terkelola Amazon

Kelola GameLift armada Amazon Anda

Gunakan GameLift konsol Amazon atau AWS CLI untuk memperbarui pengaturan armada Anda, mengubah lokasi jarak jauh, atau menghapus armada.

Memperbarui konfigurasi armada

Anda dapat memperbarui atribut armada yang dapat diubah, pengaturan port, dan konfigurasi runtime menggunakan GameLift konsol Amazon atau CLI. AWS Untuk mengubah batas penskalaan, lihat Kapasitas armada skala otomatis dengan Amazon GameLift.

Amazon GameLift console

1. Di GameLiftkonsol Amazon, 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.
     - Tambahkan atau hapus grup Metrik, yang CloudWatch digunakan Amazon untuk melacak GameLift metrik Amazon agregat untuk beberapa armada.
     - Perbarui pengaturan batas pembuatan sumber daya.
     - Aktifkan atau nonaktifkan perlindungan sesi game.
   - Konfigurasi waktu proses - Anda dapat mengubah salah satu pengaturan konfigurasi runtime berikut dan menambah atau menghapus konfigurasi waktu proses.
     - Ubah jalur Peluncuran server game Anda.
     - Tambahkan, hapus, atau ubah parameter Peluncuran opsional.
     - Ubah jumlah proses serentak yang dijalankan server game Anda.
   - Aktivasi sesi game - Ubah cara Anda ingin proses server menjalankan dan menyelenggarakan sesi game dengan memperbarui aktivasi sesi game serentak Max dan batas waktu aktivasi baru.
   - Pengaturan port EC2 - Perbarui alamat IP dan rentang port yang memungkinkan akses masuk ke armada.
4. Pilih Konfirmasi untuk menyimpan perubahan.

AWS CLI

Gunakan perintah CLI AWS berikut untuk memperbarui armada:

- update-fleet-attributes
- update-fleet-port-settings
- update-runtime-configuration

Perbarui lokasi armada

Anda dapat menambah atau menghapus lokasi jarak jauh armada menggunakan GameLift konsol Amazon atau AWS CLI. Anda tidak dapat mengubah wilayah rumah armada.

Amazon GameLift console

1. Di GameLiftkonsol Amazon, 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 terpencil baru, pilih Tambah dan pilih lokasi yang ingin digunakan instans. Daftar ini tidak termasuk contoh di mana jenis instance armada tidak tersedia.
5. Dengan lokasi baru yang dipilih, pilih Tambah. Amazon GameLift menambahkan lokasi baru ke daftar, dengan status diatur ke`NEW`. Amazon GameLift kemudian mulai menyediakan instans di setiap lokasi tambahan dan mempersiapkannya untuk menyelenggarakan sesi game.
6. Untuk menghapus lokasi terpencil yang ada dari armada, gunakan kotak centang untuk memilih satu atau beberapa lokasi yang terdaftar.
7. Dengan satu atau lebih armada dipilih, pilih Hapus. Lokasi yang dihapus tetap ada dalam daftar, dengan status disetel menjadi `DELETING`. Amazon GameLift kemudian memulai proses penghentian aktivitas di lokasi yang dihapus. Jika ada instans aktif yang menghosting sesi game, Amazon GameLift menggunakan proses penghentian server game untuk mengakhiri sesi game dengan anggun, menghentikan server game, dan mematikan instans.

AWS CLI

Gunakan perintah CLI AWS berikut untuk memperbarui lokasi armada:

- create-fleet-locations
- delete-fleet-locations

Menghapus armada

Anda dapat menghapus armada ketika Anda tidak lagi membutuhkannya. Menghapus armada secara permanen menghapus semua data yang terkait dengan sesi game dan sesi pemain, dan 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 menghapus koneksi peering VPC selama penghapusan armada.

Anda dapat menggunakan GameLift konsol Amazon atau alat AWS CLI untuk menghapus armada.

Amazon GameLift console

1. Di GameLiftkonsol Amazon, di panel navigasi, pilih Armada.
2. Pilih armada yang ingin Anda hapus. Anda hanya dapat menghapus armada dalam `ACTIVE` atau `ERROR` status.
3. Pilih Delete (Hapus).
4. Di kotak dialog Hapus armada, konfirmasikan penghapusan dengan memasukkan. **`delete`**
5. Pilih Delete (Hapus).

AWS CLI

Gunakan perintah AWS CLI berikut untuk menghapus armada:

- menghapus-armada

Tambahkan alias ke armada Amazon GameLift

GameLift Alias Amazon digunakan untuk mengabstraksi penunjukan armada. Penunjukan armada memberi tahu Amazon GameLift tempat mencari sumber daya yang tersedia saat membuat sesi permainan baru untuk pemain. Gunakan alias alih-alih ID armada tertentu untuk mengalihkan lalu

lintas pemain dengan mulus dari satu armada ke armada lainnya dengan mengubah lokasi target alias.

Ada dua tipe strategi perutean untuk alias:

- 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.

Armada memiliki umur yang terbatas, dan ada beberapa alasan untuk mengganti armada selama kehidupan permainan. Anda tidak dapat memperbarui pembuatan server game armada atau mengubah atribut sumber daya komputasi tertentu pada armada yang ada. Alih-alih, buat armada baru dengan perubahan dan kemudian alihkan pemain ke armada baru. Dengan alias, berpindah armada memiliki dampak minimal pada game Anda dan tidak terlihat oleh pemain.

Alias berguna dalam game yang tidak menggunakan antrian. Mengganti armada dalam antrean adalah masalah sederhana untuk membuat armada baru, menambahkannya ke antrean, dan menghapus armada lama, yang tidak terlihat oleh pemain. Sebaliknya, klien game yang tidak menggunakan antrian harus menentukan armada mana yang akan digunakan saat berkomunikasi dengan layanan Amazon. GameLift Tanpa alias, sakelar armada memerlukan pembaruan kode game Anda dan mungkin distribusi klien game yang diperbarui ke pemain.

Saat memperbarui fleet-id yang ditunjuk alias, ada periode transisi hingga 2 menit di mana sesi permainan pada alias mungkin berakhir di armada lama.

Buat alias baru

Anda dapat membuat alias menggunakan GameLift konsol Amazon, seperti yang dijelaskan di sini, atau dengan perintah AWS CLI create-alias.

1. Di GameLift konsol Amazon, di panel navigasi, pilih Alias.
2. Pada halaman Alias, pilih Buat alias. Sebaiknya sertakan jenis armada dalam nama alias Anda. Hal ini mempermudah proses identifikasi jenis armada saat melihat daftar alias.
3. Pada halaman Buat alias, di bawah detail Alias, lakukan hal berikut:
    a. Untuk Nama, masukkan nama alias.
    b. Untuk deskripsi, masukkan deskripsi singkat untuk identifikasi.

c. Pilih tipe perutean Sederhana atau Terminal.

4. (Opsional) Di bawah Tag, tambahkan tag ke alias dengan memasukkan pasangan Kunci dan Nilai.
5. Pilih Buat.

Mengedit alias

Anda dapat mengedit alias menggunakan GameLift konsol Amazon atau dengan perintah AWS CLI update-alias.

1. Di GameLift konsol Amazon, di panel navigasi, pilih Alias.
2. Pada halaman Alias, pilih alias yang ingin Anda edit.
3. Pada halaman alias pilih Edit.
4. Pada halaman Edit alias, Anda dapat mengedit berikut:
   - Nama Alias – Nama g mudah diingat untuk alias Anda.
   - Deskripsi – Deskripsi singkat untuk alias Anda.
   - Tipe – Strategi perutean untuk lalu lintas pemain. Pilih Sederhana untuk mengubah armada terkait atau memilih Terminal untuk mengedit pesan penghentian.
5. Pilih Simpan perubahan.

Debug masalah GameLift armada Amazon

Topik ini memberikan panduan tentang masalah konfigurasi armada untuk solusi hosting GameLift terkelola Amazon. Untuk pemecahan masalah tambahan, Anda dapat mengakses instans armada dari jarak jauh setelah armada aktif. Lihat Terhubung dari jarak jauh ke instance GameLift armada Amazon.

Masalah pembuatan armada

Saat armada dibuat, GameLift layanan Amazon memulai alur kerja yang menerapkan instance baru di setiap lokasi armada dan mempersiapkannya untuk menjalankan server game Anda. Untuk penerangan mendetail, lihat Cara kerja pembuatan GameLift armada Amazon. Armada tidak dapat meng-host sesi game dan pemain hingga mencapai status Aktif. Bagian ini membahas masalah paling umum yang menghalangi pengaktifan armada.

Mengunduh dan memvalidasi

Selama fase ini, pembuatan armada mungkin gagal jika ada masalah dengan file bangunan yang diekstrak, skrip penginstalan tidak dapat dijalankan, atau jika file yang dapat dieksekusi yang ditentukan dalam konfigurasi waktu aktif tidak disertakan dalam file bangunan. Amazon GameLift menyediakan log yang terkait dengan masing-masing masalah ini.

Jika log tidak mengungkapkan masalah, mungkin masalahnya disebabkan oleh kesalahan layanan internal. Dalam hal ini, coba buat armada lagi. Jika masalah berlanjut, pertimbangkan untuk mengunggah ulang pembuatan game (jika file rusak). Anda juga dapat menghubungi GameLift dukungan Amazon atau memposting pertanyaan di forum.

Membangun

Masalah yang menyebabkan kegagalan selama fase membangun hampir pasti karena masalah dengan file bangunan game dan/atau skrip instalasi. Verifikasi bahwa file build game Anda, seperti yang diunggah ke AmazonGameLift, dapat diinstal pada mesin yang menjalankan sistem operasi yang sesuai. Pastikan untuk menggunakan instalasi OS yang bersih, bukan lingkungan pengembangan yang ada.

Mengaktifkan

Masalah pembuatan armada yang paling umum terjadi selama faseMengaktifkan. Selama fase ini, sejumlah elemen sedang diuji, termasuk viabilitas server game, pengaturan konfigurasi runtime, dan kemampuan server game untuk berinteraksi dengan GameLift layanan Amazon menggunakan Server SDK. Masalah umum yang muncul selama aktivasi armada meliputi:

Proses server gagal dimulai.

- Pertama, periksa apakah Anda telah menetapkan jalur peluncuran dan parameter peluncuran opsional dengan benar dalam konfigurasi waktu aktif armada. Anda dapat melihat konfigurasi runtime armada saat ini menggunakan bagian Fleet detail page,Detail) atau dengan memanggil perintahAWS CLI. describe-runtime-configuration Jika konfigurasi waktu aktif terlihat benar, periksa masalah dengan file build dan/atau skrip instalasi game Anda.

Proses server dimulai tetapi armada gagal diaktifkan.

- Jika proses server mulai dan berjalan dengan sukses, tetapi armada tidak pindah ke status Aktif, kemungkinan penyebabnya adalah bahwa proses server gagal memberi tahu Amazon GameLift bahwa ia siap untuk meng-host sesi game. Periksa apakah server game Anda memanggil tindakan Server API `ProcessReady()` dengan benar (lihat Inisialisasi proses server).

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 GameLift konsol Amazon (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 Anda telah secara khusus membuat koneksi mengintip VPC untuk (lihat. VPC mengintip untuk Amazon GameLift Skenario ini terjadi karena proses penghapusan armada juga termasuk menghapus VPC armada dan koneksi peering VPC apa pun. Anda harus terlebih dahulu mendapatkan otorisasi dengan memanggil Amazon GameLift service API CreateVpcPeeringAuthorization() atau menggunakan perintah AWS

CLI. `create-vpc-peering-authorization` Setelah Anda memiliki otorisasi, Anda dapat menghapus armada.

Masalah armada Realtime Servers

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 Servers 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 memanggil `processEnding` skrip Realtime Anda dengan sukses. Meskipun armada berjalan aktif dan sesi game dimulai, tidak ada metode untuk menghentikannya. Akibatnya, Realtime Servers yang menjalankan sesi game tidak pernah dibebaskan untuk memulai yang baru, dan sesi game baru hanya dapat dimulai ketika Realtime Servers baru diputar. Selain itu, pembaruan skrip Realtime tidak berdampak pada sesi game yang sudah berjalan, hanya satu.

Untuk mencegah hal ini terjadi, skrip perlu menyediakan mekanisme untuk memicu panggilan `processEnding`. Seperti yang diilustrasikan dalam Contoh skrip Server Realtime, 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 mengalami skenario ini, ada beberapa solusi untuk membuat Realtime Servers Anda tidak macet. Triknya adalah memicu proses Realtime Servers—atau instans armada yang mendasarinya—untuk memulai ulang. Dalam acara ini, GameLift secara otomatis menutup sesi permainan untuk Anda. Setelah Realtime Servers dibebaskan, mereka dapat memulai sesi game baru menggunakan skrip Realtime versi terbaru.

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 game yang ada, dan membiarkan Anda memulai dengan skrip Realtime terbaru yang 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 instance GameLift armada Amazon.

Jika Anda memilih untuk tidak menyertakan cara untuk memanggil `processEnding` dalam skrip Realtime Anda, ada beberapa situasi rumit yang mungkin terjadi bahkan ketika armada berjalan aktif dan sesi game 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 Servers tidak mengambil pembaruan skrip.

Terhubung dari jarak jauh ke instance GameLift armada Amazon

Anda dapat terhubung ke instans apa pun di armada EC2 GameLift terkelola Amazon aktif Anda. Alasan umum untuk mengakses instance 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 instance dalam armada aktif. Armada non-aktif, yang mengaktifkan atau berada dalam keadaan kesalahan, mungkin dapat diakses untuk waktu yang singkat. Untuk bantuan terkait masalah aktivasi armada, lihat Debug masalah GameLift armada Amazon.
- Menghubungkan ke instans aktif tidak memengaruhi aktivitas hosting instans. Instance terus memulai dan menghentikan proses server berdasarkan konfigurasi runtime. Ini mengaktifkan dan menyelenggarakan sesi permainan. Ini mungkin ditutup sebagai tanggapan terhadap peristiwa penurunan skala atau acara 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 GameLift layanan Amazon.

Kumpulkan data instance

Kumpulkan informasi berikut:

- ID dari instance yang ingin Anda sambungkan. Anda dapat menggunakan ID instance atau ARN.
- Versi SDK GameLift server Amazon yang digunakan pada instance. Server SDK terintegrasi dengan game build yang berjalan pada instance.

Untuk mengambil data instance

Langkah-langkah berikut mengasumsikan Anda memiliki ID armada EC2 terkelola untuk instance yang ingin Anda sambungkan.

1. Dapatkan nama komputasi.

   Call list-compute untuk armada EC2 terkelola untuk mendapatkan daftar semua komputasi aktif dalam armada. Untuk armada satu lokasi, tentukan ID armada atau ARN. Untuk armada multi-lokasi, tentukan ID armada atau ARN dan lokasi. Untuk armada EC2 terkelola, komputasi adalah instans EC2 dan properti yang dikembalikan `ComputeName` adalah ID instans. Sebagai contoh:

   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":
    "b08444ki909kvqu6zpw3is24x5pyz4b6m05i3jbxvpk9craztu0lqrbbrbnbkks.uwp57060n1k6dnlnw49b78hg1
   west-2.amazongamelift.com",
         "ComputeStatus": "Active",
         "Location": "sa-east-1",
   ```

```
      "CreationTime": "2023-07-09T22:51:45.931000-07:00",
      "OperatingSystem": "AMAZON_LINUX",
      "Type": "c4.large"
    }
  ]
}
```

2. Temukan versi SDK server.

   Versi SDK server adalah atribut sumber daya build.

   a. Hubungi describe-fleet-attributesdengan ID armada 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_2",
  "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, gunakan petunjuk berikut untuk menyambung ke instans menggunakan Amazon EC2 Systems Manager (SSM). Anda dapat mengakses instans jarak jauh yang menjalankan Windows atau Linux.

1. Minta kredensyal akses untuk instance. Bila Anda memiliki nama komputasi dan ID armada untuk instance yang ingin Anda sambungkan, panggil get-compute-access. Jika berhasil, Amazon GameLift mengembalikan satu set kredensyal 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 kredensyal akses. Anda dapat secara opsional mengekspor kredensyal ke variabel lingkungan dan menggunakannya untuk mengonfigurasi 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, lihat Memulai sesi (AWSCLI) di Panduan Pengguna Amazon EC2 Systems Manager. Gunakan kredensyal yang Anda peroleh di Langkah 1. Sebagai contoh:

```
aws ssm start-session \
--target i-11111111a222b333c \
--region us-west-2
```

Connect ke 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 kredensyal akses untuk instance. Bila Anda memiliki ID instance, gunakan perintah get-instance-accessuntuk meminta kredensyal akses. Jika berhasil, Amazon GameLift mengembalikan sistem operasi instans, alamat IP, dan satu set kredensyal (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. Kredenensi 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 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-----
nEXAMPLEKEYKCAQEAy7WZhaDsrA1W3mRlQtvhwyORRX8gnxgDAfRt/gx42kWXsT4rXE/b5CpSgie/
\nvBoU7jLxx92pNHoFnByP+Dc21eyyz6CvjTmWA0JwfWiW5/akH7iO5dSrvC7dQkW2duV5QuUdE0QW
\nZ/aNxMniGQE6XAgfwlnXVBwrerrQo+ZWQeqiUwwMkuEbLeJFLhMCvYURpUMSC1oehm449ilx9X1F
\nG50TCFeOzfl8dqqCP6GzbPaIjiU19xX/azOR9V+tpUOzEL+wmXnZt3/nHPQ5xvD2OJH67km6SuPW
\noPzev/D8V+x4+bHthfSjR9Y7DvQFjfBVwHXigBdtZcU2/wei8D/HYwIDAQABAoIBAGZ1kaEvnrqu
\n/uler7vgIn5m7lN5LKw4hJLAIW6tUT/fzvtcHK0SkbQCQXuriHmQ2MQyJX/0kn2NfjLV/
ufGxbL1\nmb5qwMGUnEpJaZD6QSSs3kICLwWUYUiGfc0uiSbmJoap/
GTLU0W5Mfcv36PaBUNy5p53V6G7hXb2\nbahyWyJNfjLe4M86yd2YK3V2CmK+X/
BOsShnJ36+hjrXPPWmV3N9zEmCdJjA+K15DYmhm/
tJWSD9\n81oGk9TopEp7CkIfatEATyyZiVqoRq6k64iuM9JkA3OzdXzMQexXVJ1TLZVEH0E7bhlY9d8O1ozR
\noQs/FiZNAx2iijCWyv0lpjE73+kCgYEA9mZtyhkHkFDpwrSM1APaL8oNAbbjwEy7Z5Mqfql
+lIp1\nYkriL0DbLXlvRAH+yHPRit2hHOjtUNZh4Axv+cpg09qbUI3+43eEy24B7G/Uh
+GTfbjsXsOxQx/x\np9otyVwc7hsQ5TA5PZb
+mvkJ5OBEKzet9XcKwONBYELGhnEPe7cCgYEA06Vgov6YHleHui9kHuws
\nayav0elc5zkxjF9nfHFJRry21R1trw2Vdpn+9g481URrpzWVOEihvm+xTtmaZlSp//lkq75XDwnU
\nWA8gkn6O3QE3fq2yN98BURsAKdJfJ5RL1HvGQvTe10HLYYXpJnEkHv+Unl2ajLivWUt5pbBrKbUC
\ngYBjbO+OZk0sCcpZ29sbzjYjpIddErySIyRX5gV2uNQwAjLdp9PfN295yQ+BxMBXiIycWVQiw0bH
\noMo7yykABY7Ozd5wQewBQ4AdSlWSX4nGDtsiFxWiI5sKuAAeOCbTosy1s8w8fxoJ5Tz1sdoxNeGs
\nArq6Wv/G16zQuAE9zK9vvwKBgF+09VI/1wJBirsDGz9whVWfFPrTkJNvJZzYt69qezxlsjgFKshy
\nWBhd4xHZtmCqpBPlAymEjr/TOlbxyARmXMnIOWIAnNXMGB4KGSyl1mzSVAoQ+fqR+cJ3d0dyPl1j
\njjb0Ed/NY8frlNDxAVHE8BSkdsx2f6ELEyBKJSRr9snRAoGAMrTwYneXzvTskF/S5Fyu0iOegLDa
\nNWUH38v/nDCgEpIXD5Hn3qAEcju1IjmbwlvtW+nY2jVhv7UGd8MjwUTNGItdb6nsYqM2asrnF3qS
\nVRkAKKKYeGjkpUfVTrW0YFjXkfcrR/V+QFL5OndHAKJXjW7a4ejJLncTzmZSpYzwApc=\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 GameLift armada Amazon melalui port apa pun yang diotorisasi 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 ke Amazon GameLift. 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.
- Sumber daya GameLift hosting Amazon. Direktori root `Whitewater` berisi file yang digunakan oleh GameLift layanan Amazon 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 UpdateRuntimeConfigurationatau). 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/`

Penskalaan kapasitas GameLift hosting Amazon

Kapasitas hosting, diukur dalam instans, mewakili jumlah sesi game yang GameLift dapat dihosting Amazon secara bersamaan dan jumlah pemain bersamaan yang dapat diakomodasi oleh sesi game 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: wilayah asal AWS armada. Saat melihat atau menskalakan kapasitas, informasi tersebut tercantum berdasarkan lokasi, termasuk Wilayah asal armada dan lokasi terpencil tambahan.

Anda dapat mengatur jumlah instance yang akan dipelihara 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.

Topik dalam bahagian ini memberikan bantuan terperinci dengan tugas-tugas berikut:

- Tetapkan batas minimum dan maksimum untuk penskalaan kapasitas
- Secara manual mengatur tingkat kapasitas
- Gunakan penskalaan otomatis berbasis target
- Mengelola penskalaan otomatis berbasis aturan (fitur lanjutan)
- Nonaktifkan sementara penskalaan otomatis

Anda dapat melakukan sebagian besar aktivitas penskalaan armada menggunakan konsol AmazonGameLift. Anda juga dapat menggunakan AWS SDK atau AWS Command Line Interface (AWS CLI) dengan API GameLift layanan Amazon.

Untuk mengelola kapasitas armada di konsol

1. Buka GameLiftkonsol Amazon.
2. Di panel navigasi, pilih Hosting, Armada.

3. Di halaman Armada, pilih nama armada aktif untuk membuka halaman detail armada.
4. Pilih tab Scaling. 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 tunda aktivitas penskalaan otomatis untuk setiap lokasi.

Topik

Tetapkan batas GameLift kapasitas Amazon

Saat menskalakan kapasitas hosting untuk lokasi GameLift armada Amazon, 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, batasan 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 untuk meningkatkan lokasi armada untuk memenuhi permintaan pemain tetapi mencegah biaya hosting pelarian, seperti selama serangan DDOS. Siapkan CloudWatchalarm 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 GameLiftkonsol Amazon atau dengan menggunakan AWS Command Line Interface (AWS CLI).

Mengatur batas kapasitas

Console

1. Buka GameLiftkonsol Amazon.
2. Di panel navigasi, pilih Hosting, Armada.
3. Di 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. Di kotak dialog Edit scaling capacity, atur jumlah instance 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-capacityperintah dengan ID armada dan lokasi yang ingin Anda ubah kapasitasnya. Perintah ini mengembalikan FleetCapacityobjek yang mencakup 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-capacityperintah 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 mengembalikan ID armada. Jika `min-size` nilai baru `max-size` atau bertentangan dengan `desired-instances` setelan saat ini, Amazon GameLift mengembalikan kesalahan.

Mengatur kapasitas GameLift armada Amazon secara manual

Saat Anda membuat armada baru, Amazon GameLift secara otomatis menetapkan instans yang diinginkan ke satu instans di setiap lokasi armada. Kemudian, Amazon GameLift menerapkan satu instance baru di setiap lokasi. Untuk mengubah kapasitas armada, Anda dapat menambahkan kebijakan penskalaan otomatis berbasis target, atau Anda dapat mengatur jumlah instance yang diinginkan untuk suatu lokasi secara manual. Untuk informasi selengkapnya, lihat Penskalaan kapasitas armada.

Mengatur 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 ini segera menyetel ulang kapasitas yang diinginkan berdasarkan aturan penskalaannya sendiri.

Anda dapat mengatur kapasitas secara manual di GameLift konsol Amazon 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 instance 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 GameLiftkonsol Amazon.
2. Di panel navigasi, pilih Hosting, Armada.
3. Di halaman Armada, pilih nama armada aktif untuk membuka halaman detail armada.
4. Pada tab Penskalaan, di bawah Lokasi penskalaan otomatis 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 pilihan Anda untuk instans yang diinginkan, lalu pilih Konfirmasi. Ini memberi tahu Amazon GameLift jumlah instans yang harus dipertahankan dalam keadaan aktif, siap untuk menjadi tuan rumah sesi game.

Amazon GameLift merespons perubahan dengan menerapkan instans tambahan atau mematikan instans yang tidak dibutuhkan. Saat Amazon GameLift menyelesaikan proses ini, jumlah instans 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-capacityperintah dengan ID armada dan lokasi yang ingin Anda ubah kapasitasnya. Perintah ini mengembalikan FleetCapacityobjek yang mencakup 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-capacityperintah 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 mengembalikan ID armada. Jika pengaturan instans baru yang diinginkan berada di luar batas minimum dan maksimum, Amazon GameLift mengembalikan kesalahan.

Kapasitas armada skala otomatis dengan Amazon GameLift

Gunakan penskalaan otomatis di Amazon GameLift untuk menskalakan kapasitas armada Anda secara dinamis sebagai respons terhadap aktivitas server game. Saat pemain tiba dan memulai sesi permainan, penskalaan otomatis dapat menambahkan lebih banyak instance; saat permintaan pemain berkurang, penskalaan otomatis dapat mengakhiri instance yang tidak dibutuhkan. Penskalaan otomatis adalah cara 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 kapan harus menaikkan atau menurunkan skala. Ada dua jenis kebijakan penskalaan: berbasis target dan berbasis aturan. Pendekatan berbasis target — pelacakan target —adalah solusi lengkap. Kami merekomendasikannya sebagai pilihan paling sederhana dan paling efektif. Kebijakan penskalaan berbasis aturan mengharuskan Anda 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 GameLift konsol Amazon, AWS Command Line Interface (AWS CLI), atau SDK. AWS Anda dapat mengelola penskalaan otomatis berbasis aturan hanya menggunakan AWS SDK AWS CLI atau hanya, meskipun Anda dapat melihat kebijakan penskalaan berbasis aturan di konsol.

Topik

Penskalaan otomatis berbasis target

Penskalaan otomatis berbasis target untuk Amazon GameLift menyesuaikan tingkat kapasitas berdasarkan metrik armada. `PercentAvailableGameSessions` 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 mendapatkan 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 Anda inginkan untuk mempertahankan armada. Karena `PercentAvailableGameSessions` mengukur persentase sumber daya yang tersedia, ukuran buffer sebenarnya adalah persentase dari total kapasitas armada. Amazon GameLift menambahkan atau menghapus instans 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.

Mengatur penskalaan otomatis berbasis target

Console

1. Buka GameLiftkonsol Amazon.
2. Di panel navigasi, pilih Hosting, Armada.
3. Di halaman Armada, pilih nama armada aktif untuk membuka halaman detail armada.
4. Pilih tab Scaling. Tab ini menampilkan metrik penskalaan historis armada dan berisi kendali untuk menyesuaikan pengaturan penskalaan saat ini.
5. Di bawah kapasitas Scaling, periksa apakah ukuran Min dan batas ukuran Max sesuai untuk armada. Dengan penskalaan otomatis diaktifkan, kapasitas menyesuaikan antara dua batas ini.
6. Dalam kebijakan penskalaan otomatis berbasis target, pilih Edit.

7. Di kotak dialog Edit kebijakan penskalaan otomatis berbasis target, untuk sesi game Persen yang tersedia, tetapkan persentase yang ingin Anda pertahankan, lalu pilih Konfirmasi. Setelah Anda mengonfirmasi pengaturan, Amazon GameLift menambahkan kebijakan berbasis target baru berdasarkan kebijakan penskalaan otomatis berbasis Target.

AWS CLI

1. Tetapkan batas kapasitas. Tetapkan nilai batas menggunakan update-fleet-capacityperintah. Untuk informasi selengkapnya, lihat Tetapkan batas GameLift kapasitas Amazon.
2. Buat kebijakan baru. Buka jendela baris perintah dan gunakan put-scaling-policyperintah 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 di Amazon GameLift memberikan kontrol berbutir halus saat melakukan 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 penskalaan yang merespons. Kebijakan berbasis aturan berguna untuk melengkapi penskalaan berbasis target untuk menangani keadaan khusus.

Kebijakan berbasis aturan menyatakan hal berikut: “Jika metrik armada memenuhi atau melintasi nilai ambang batas untuk jangka waktu tertentu, maka ubah kapasitas armada dengan jumlah yang ditentukan.” 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 API layanan Amazon GameLift. Anda dapat melihat semua kebijakan aktif di GameLift konsol Amazon.

Untuk menghentikan sementara semua kebijakan penskalaan armada, gunakan perintah. AWS CLI 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-capacityperintah. Untuk informasi selengkapnya, lihat Tetapkan batas GameLift kapasitas Amazon.
2. Buat kebijakan baru. Buka jendela baris perintah dan gunakan put-scaling-policyperintah 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-policyperintah 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>*, lalu ubah kapasitas armada menggunakan *<adjustment type>* ke/oleh *<adjustment value>*.

Misalnya, pernyataan kebijakan ini memulai peristiwa peningkatan skala setiap kali kapasitas ekstra armada kurang dari yang diperlukan untuk menangani 50 sesi permainan baru:

Jika `AvailableGameSessions` tetap di`less than 50` untuk `10 minutes`, maka ubah kapasitas armada menggunakan `ChangeInCapacity`dengan `1 instances`.

Nama metrik

Untuk memulai peristiwa penskalaan, tautkan kebijakan penskalaan otomatis ke salah satu metrik khusus armada berikut. Untuk deskripsi metrik lengkap, lihatGameLiftMetrik Amazon 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 game, Anda dapat menggunakan metrik berikut:

- Kedalaman antrian - Jumlah permintaan sesi game yang tertunda untuk 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 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 akan 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 peristiwa 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 harus 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 dengan 10 instance, 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 tingkatkan 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 memaksimalkan 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 dalam menggunakan beberapa kebijakan.

Dengan beberapa kebijakan, setiap kebijakan berperilaku independen. Tidak ada cara untuk mengontrol urutan kejadian 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 memulai satu sama lain. Misalnya, Anda dapat membuat loop tak terbatas jika Anda membuat kebijakan penskalaan dan penskalaan 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 kisaran 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 maksimum.

Kapasitas armada juga dibatasi oleh batasan jenis instans armada dan oleh 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 menunggu 10 menit sebelum merespons pemicu dari kebijakan yang sama. Penantian ini memberi GameLift waktu Amazon untuk menambahkan instans baru, meluncurkan server game, menghubungkan pemain, dan mulai mengumpulkan data dari instans baru. Selama waktu ini, Amazon GameLift mengevaluasi kebijakan terhadap metrik dan melacak periode evaluasi kebijakan, yang dimulai ulang setelah peristiwa penskalaan terjadi. Ini berarti bahwa kebijakan penskalaan dapat memulai kejadian penskalaan lain segera setelah waktu tunggu selesai.

Tidak ada waktu tunggu antara kejadian penskalaan yang dimulai kebijakan penskalaan otomatis yang berbeda.

Menyiapkan GameLift antrean Amazon untuk penempatan sesi game

Antrian sesi permainan adalah mekanisme utama untuk memproses permintaan sesi game baru dan menemukan server game yang tersedia untuk menjadi tuan rumah mereka. Antrian menawarkan manfaat yang signifikan bagi pengembang game dan pemain. Ini termasuk:

- Antrian memberikan penempatan terbaik. Saat memproses permintaan penempatan sesi game, antrean menggunakan GameLift algoritme Amazon untuk memprioritaskan lokasi antrean berdasarkan serangkaian preferensi yang ditentukan.
- Host game di armada Spot dengan harga lebih rendah. Gunakan antrean untuk mengoptimalkan penggunaan armada Spot AWS, yang menawarkan biaya hosting yang jauh lebih rendah. Secara default, antrian selalu mencoba menempatkan sesi permainan baru di armada Spot.
- Tempatkan game baru lebih cepat selama permintaan tinggi. Antrian menggunakan beberapa lokasi yang mungkin untuk penempatan. Ini berarti bahwa selalu ada kapasitas fallback jika lokasi penempatan yang diinginkan tidak tersedia.
- Jadikan ketersediaan game lebih tangguh. Pemadaman bisa terjadi. Dengan antrean multi-lokasi, perlambatan atau pemadaman tidak harus memengaruhi akses pemain ke game Anda.
- Gunakan kapasitas armada ekstra dengan lebih efisien. Untuk menangani lonjakan permintaan pemain yang tidak terduga, antrian menyediakan akses cepat ke kapasitas hosting tambahan. Lokasi armada dalam antrian menyediakan kapasitas cadangan satu sama lain. Lokasi meningkat atau menurun skalanya berdasarkan permintaan pemain.

- Dapatkan metrik tentang penempatan sesi game dan performa antrian. Amazon GameLift memancarkan metrik antrian, termasuk statistik keberhasilan dan kegagalan penempatan, jumlah permintaan dalam antrian, dan waktu rata-rata yang meminta pengeluaran dalam antrian. Anda dapat melihat metrik ini di GameLift konsol Amazon atau diCloudWatch.

Untuk memulai antrian, lihat. Desain antrean sesi game

Desain antrean sesi game

Topik ini menjelaskan cara merancang antrian yang memberikan pengalaman pemain dengan latensi minimal dan yang secara efisien menggunakan sumber daya hosting. Untuk informasi lebih lanjut tentang antrian sesi permainan dan cara kerjanya, lihat. Menyiapkan GameLift antrean Amazon untuk penempatan sesi game

GameLiftFitur Amazon ini memerlukan antrean:

- Perjodohan dengan FlexMatch
- Menggunakan Instans Spot dengan Amazon GameLift

Tentukan cakupan antrean Anda

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 game sendiri.

Untuk mengatur penempatan sesi game untuk populasi pemain Anda, buat antrian terpisah untuk setiap segmen pemain. Lingkup setiap antrian untuk menempatkan pemain ke server game yang benar. Beberapa cara umum untuk lingkup antrian meliputi:

- Dengan lokasi geografis. Saat menerapkan server game Anda di beberapa area geografis, Anda dapat membuat antrean untuk pemain di setiap lokasi untuk mengurangi latensi pemain.
- Dengan membangun atau skrip variasi. 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.
- Menurut jenis acara. Anda dapat membuat antrean khusus untuk mengelola game bagi peserta dalam turnamen atau kejadian khusus lainnya.

Buat kebijakan latensi pemain

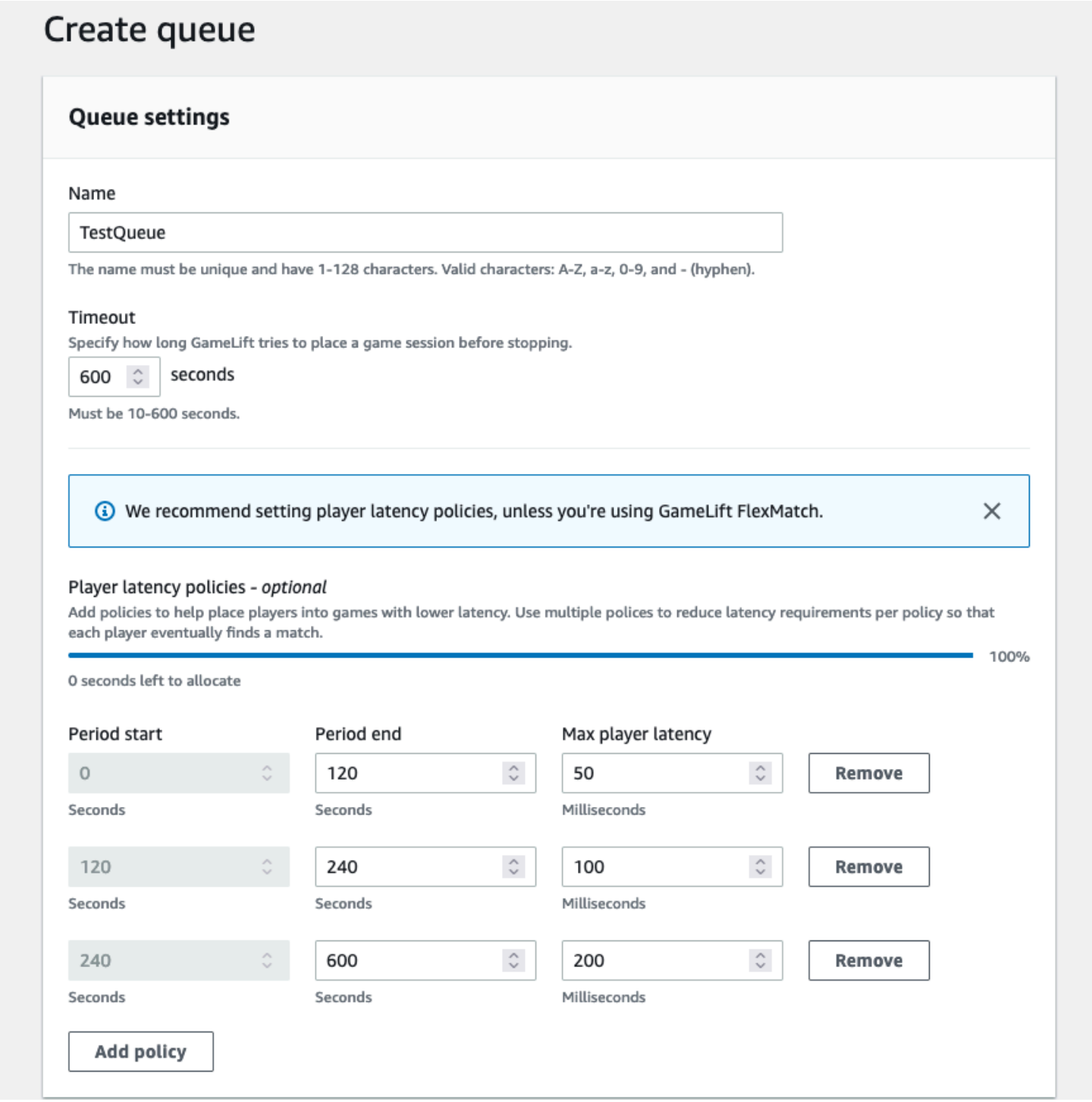
Jika permintaan penempatan Anda menyertakan data latensi pemain, algoritme menemukan sesi game di lokasi dengan latensi rata-rata terendah untuk semua pemain. Menempatkan sesi game berdasarkan latensi pemain rata-rata GameLift mencegah Amazon menempatkan sebagian besar pemain dalam game dengan latensi tinggi. Namun, Amazon GameLift masih menempatkan pemain dengan latensi ekstrem. Untuk mengakomodasi pemain ini, buat kebijakan latensi pemain.

Kebijakan latensi pemain GameLift mencegah Amazon menempatkan sesi permainan yang diminta di mana saja pemain dalam permintaan akan mengalami latensi melebihi nilai maksimum. Kebijakan latensi pemain juga dapat GameLift mencegah Amazon mencocokkan permintaan sesi game dengan pemain latensi yang lebih tinggi.

> **Tip**
>
> Untuk mengelola aturan spesifik latensi, seperti memerlukan latensi serupa di semua pemain dalam grup, Anda dapat menggunakan Amazon GameLift FlexMatch untuk membuat aturan perjodohan berbasis latensi.

Misalnya, pertimbangkan antrean ini dengan batas waktu tunggu 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.



Membuat antrean 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 dalam sesi permainan dengan latensi minimal.

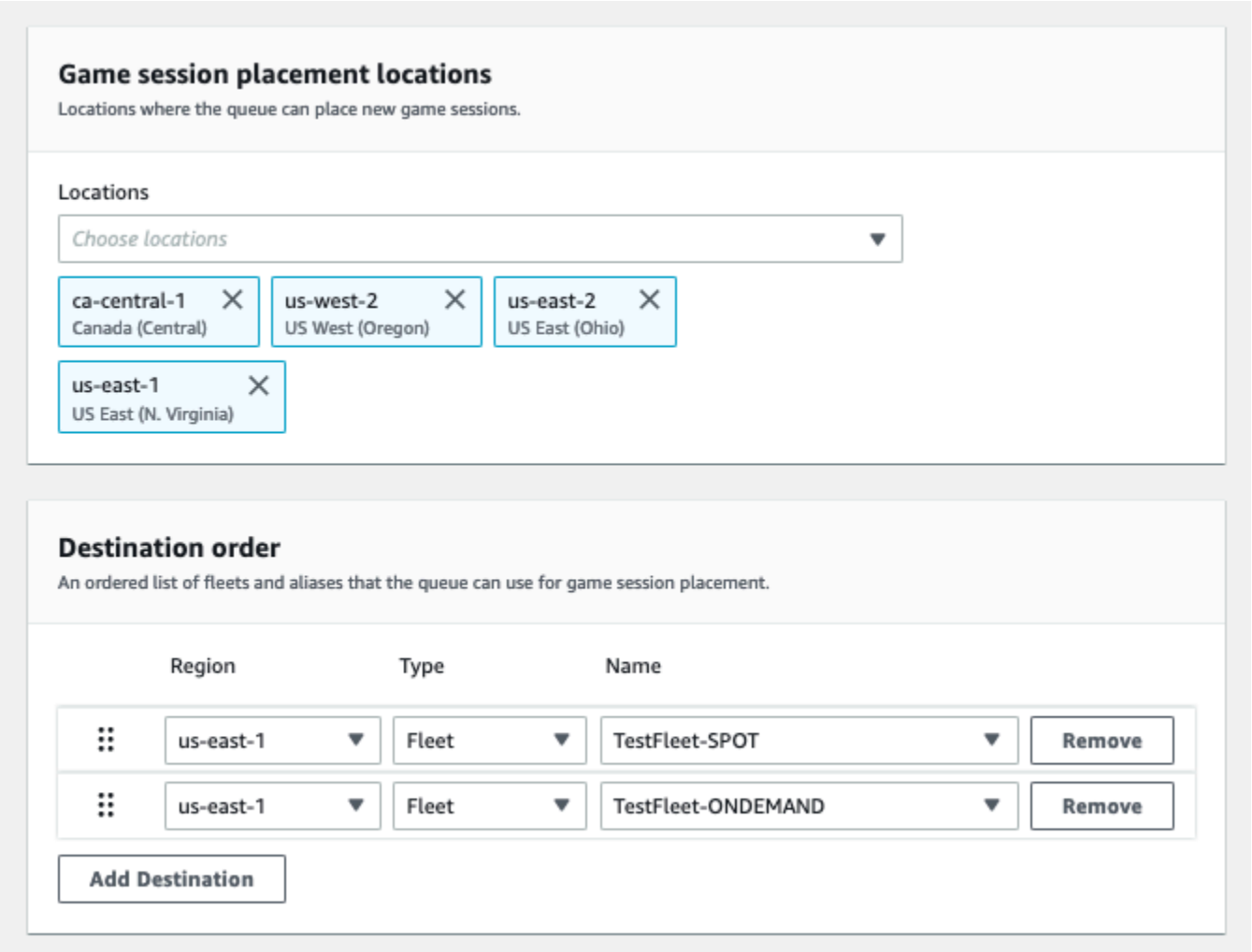
Jika Anda membuat antrean multi-lokasi yang menggunakan armada Instans Spot, ikuti petunjuk di dalamnya. Tutorial: Mengatur antrean sesi game untuk Instans Spot

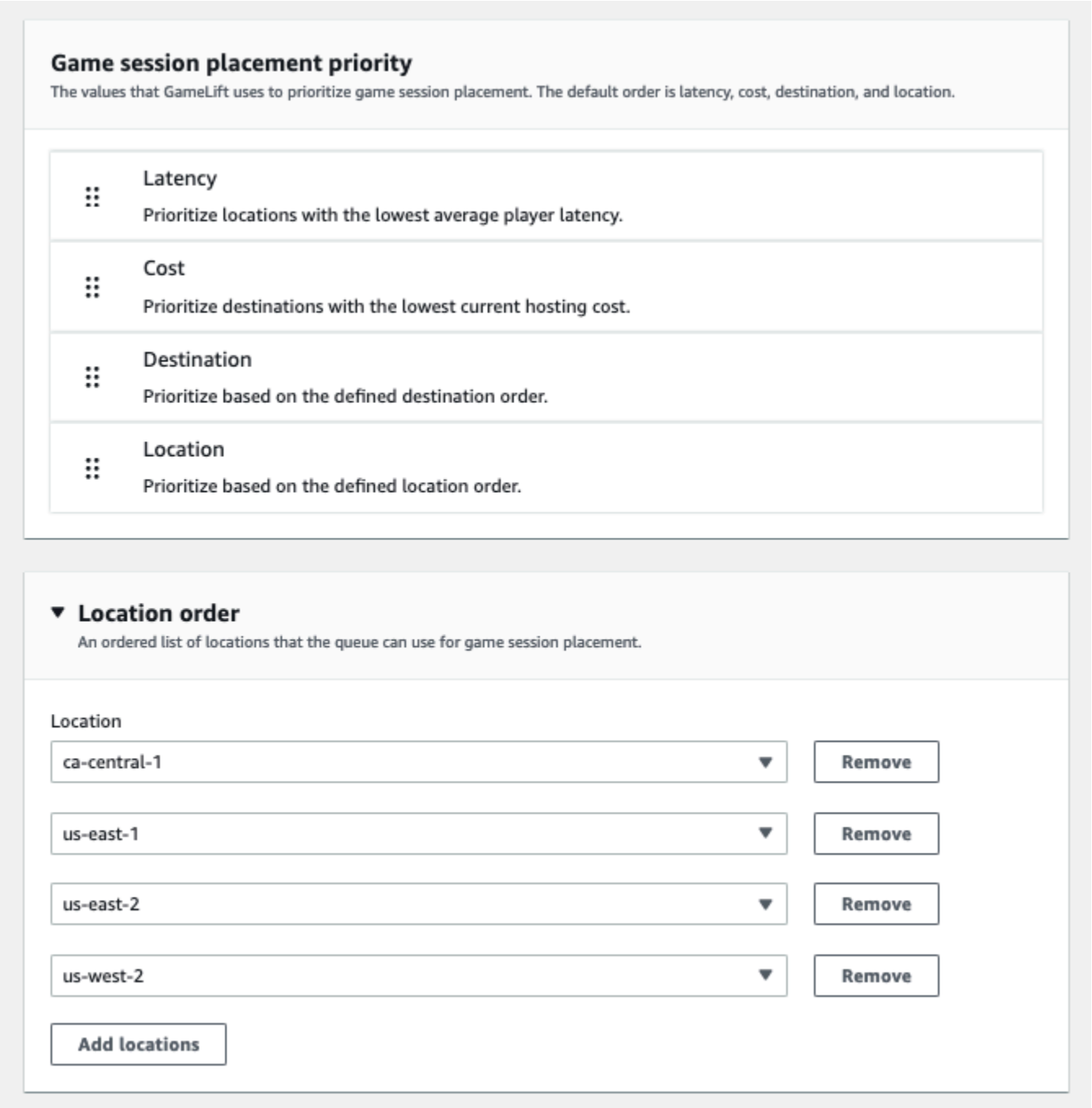
Salah satu cara untuk membuat antrian multi-lokasi adalah dengan menambahkan armada multi-lokasi ke antrian. Dengan begitu, antrian dapat menempatkan sesi permainan di salah satu lokasi armada. Anda juga dapat menambahkan armada lain dengan konfigurasi yang berbeda atau lokasi rumah untuk redundansi. 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 Spot Instance dan satu armada On-Demand Instance. Setiap armada memiliki 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 menempatkan sesi game. Dalam contoh ini, kita mencantumkan armada Spot Instance terlebih dahulu dan armada On-Demand Instance kedua.
3. Tentukan urutan prioritas penempatan sesi permainan antrian. Urutan ini menentukan di mana antrian mencari pertama untuk server game yang tersedia. Dalam contoh ini, kita menggunakan urutan prioritas default.
4. Tentukan urutan lokasi. Jika Anda tidak menentukan urutan lokasi, Amazon GameLift menggunakan 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



Prioritaskan penempatan sesi game

Amazon GameLift menggunakan algoritme FleetiQ untuk menentukan di mana menempatkan sesi permainan baru berdasarkan serangkaian kriteria yang diurutkan. Anda dapat menggunakan urutan prioritas default, atau Anda dapat menyesuaikan pesanan.

Urutan prioritas default

Untuk permintaan penempatan yang menyertakan data latensi pemain, FleetiQ memprioritaskan kriteria penempatan sesi game dalam urutan default berikut:

1. Latensi - Latensi rata-rata terendah untuk semua pemain dalam permintaan.
2. Biaya — Biaya hosting terendah, jika latensi sama di beberapa lokasi. Biaya hosting terutama didasarkan pada kombinasi jenis instans dan lokasi.
3. Tujuan - Urutan tujuan, jika latensi dan biaya sama di beberapa lokasi. FleetiQ memprioritaskan tujuan berdasarkan urutan yang tercantum dalam konfigurasi antrian.
4. Lokasi - Urutan lokasi, jika latensi, biaya, dan tujuan sama di beberapa lokasi. FleetiQ memprioritaskan lokasi berdasarkan urutan yang tercantum dalam konfigurasi antrian.

Urutan prioritas khusus

Untuk menyesuaikan urutan prioritas antrean di GameLiftkonsol Amazon, seret nilai prioritas ke posisi yang Anda inginkan. Untuk menyesuaikan urutan prioritas antrian menggunakan AWS Command Line Interface (AWS CLI), gunakan create-game-session-queueperintah dengan opsi. --priority-configuration Anda dapat menggunakan perintah ini untuk membuat antrian baru atau memperbarui antrian yang ada.

Algoritma FleetiQ menambahkan kriteria apa pun yang tidak disebutkan secara eksplisit ke akhir daftar Anda, berdasarkan urutan default. Jika Anda menyertakan kriteria lokasi dalam konfigurasi prioritas Anda, Anda juga harus menyediakan daftar lokasi yang diurutkan.

Desain beberapa antrean sesuai keperluan

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 salah satu server game antrian.
- Kelompok pemain yang berbeda. Anda dapat menyesuaikan cara Amazon GameLift menempatkan sesi game 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 game. Anda dapat mengatur antrian berdasarkan cara mengumpulkan metrik penempatan sesi game. Untuk informasi selengkapnya, lihat GameLiftMetrik Amazon untuk antrean.

Evaluasi metrik antrean

Gunakan metrik untuk mengevaluasi seberapa baik performa antrean Anda. Anda dapat melihat metrik yang terkait dengan antrian di GameLiftkonsol Amazon atau di Amazon. CloudWatch Untuk daftar dan deskripsi metrik antrian, lihat. GameLiftMetrik Amazon untuk antrean

Metrik antrian dapat memberikan wawasan tentang hal-hal berikut:

- Performa antrian keseluruhan — Metrik antrean menunjukkan seberapa sukses antrean merespons permintaan penempatan. Metrik ini juga dapat membantu Anda mengidentifikasi kapan dan mengapa penempatan gagal. Untuk antrian dengan armada skala manual, `AverageWaitTime` dan `QueueDepth` metrik dapat menunjukkan kapan Anda harus menyesuaikan kapasitas untuk antrian.
- Kinerja algoritme FleetiQ - Untuk permintaan penempatan menggunakan algoritme FleetiQ, metrik menunjukkan seberapa sering algoritme menemukan penempatan sesi game 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 tidak GameLift dapat menemukan penempatan yang ideal. Untuk informasi lebih lanjut tentang metrik, lihat Pantau Amazon GameLift dengan Amazon CloudWatch.
- Penempatan spesifik lokasi - Untuk antrean multi-lokasi, metrik menunjukkan penempatan yang berhasil berdasarkan lokasi. Untuk antrian yang menggunakan algoritma FleetiQ, data ini memberikan wawasan yang berguna tentang di mana aktivitas pemain terjadi.

Saat mengevaluasi metrik untuk kinerja algoritma FleetiQ, pertimbangkan tips berikut:

- Untuk melacak tingkat antrian menemukan penempatan yang ideal, gunakan metrik yang dikombinasikan dengan `PlacementsSucceeded` metrik FleetiQ untuk latensi terendah dan harga terendah.
- Untuk meningkatkan tingkat antrian 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 Instans Spot dianggap tidak layak jika tingkat interupsi untuk jenis instans tertentu terlalu tinggi. Untuk mengatasi masalah ini, ubah antrean untuk menggunakan armada Instans Spot dengan jenis instans yang berbeda. Kami menyarankan Anda menyertakan armada Instans Spot dengan jenis instans yang berbeda di setiap lokasi.

Praktik terbaik untuk antrean sesi GameLift game Amazon

Berikut adalah beberapa praktik terbaik yang dapat membantu Anda membangun antrean sesi game yang efektif untuk penempatan sesi game.

Praktik terbaik untuk antrean dengan jenis armada apa pun

Sebuah antrean berisi daftar tujuan armada di mana sesi game baru dapat ditempatkan. Setiap armada dapat memiliki instans yang di-deploy di beberapa lokasi geografis. Saat memilih penempatan, antrean memilih kombinasi armada dan lokasi armada. Anda menyediakan satu set prioritas untuk antrean untuk digunakan ketika memilih penempatan.

Pertimbangkan panduan dan praktik terbaik berikut ini:

- Tambahkan armada di lokasi yang mencakup pemain Anda. Anda dapat menambahkan armada dan alias di lokasi 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 game Anda. Sebuah antrean memprioritaskan pilihan penempatan berdasarkan beberapa elemen, termasuk urutan daftar tujuan.

- Buat antrean 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 game di armada Anda harus memiliki protokol komunikasi yang kompatibel.

Praktik terbaik untuk antrean dengan armada Spot

Jika antrean Anda mencakup armada Spot, siapkan antrean yang tangguh. Ini memanfaatkan penghematan biaya dengan armada Spot sekaligus meminimalkan efek interupsi sesi game. Untuk bantuan dengan membangun armada dan antrian sesi permainan dengan benar untuk digunakan dengan armada Spot, lihat. Tutorial: Mengatur antrean sesi game untuk Instans Spot Untuk informasi selengkapnya tentang instans Spot, lihatMenggunakan Instans Spot dengan Amazon GameLift.

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 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 GameLiftkonsol Amazon untuk melihat data harga historis untuk jenis instans.

Buat antrean sesi game

Antrean digunakan untuk menempatkan sesi game baru dengan sumber daya hosting terbaik yang tersedia di beberapa armada dan wilayah. Untuk mempelajari selengkapnya tentang membangun antrean untuk game Anda, lihat Desain antrean sesi game.

Dalam client game, sesi game baru dimulai dengan antrean dengan menggunakan permintaan penempatan. Pelajari selengkapnya tentang penempatan sesi game di Buat sesi permainan.

Saat memperbarui tujuan antrian dalam antrian, ada periode transisi singkat (hingga 30 detik) di mana sesi permainan yang ditempatkan di tujuan antrian mungkin masih berakhir di armada lama.

Console

1. Di GameLiftkonsol Amazon, di halaman navigasi, pilih Antrian.
2. Pada halaman Antrian, pilih Buat antrian.
3. Pada halaman Buat antrian, di bawah pengaturan antrian lakukan hal berikut:
   a. Untuk Nama, masukkan nama antrian.
   b. Untuk Timeout, masukkan lama Anda GameLift ingin Amazon mencoba menempatkan sesi permainan sebelum berhenti. Amazon GameLift mencari sumber daya yang tersedia di armada apa pun hingga permintaan habis.
   c. (Opsional) Untuk kebijakan latensi Pemain, masukkan berapa lama Amazon GameLift harus mencari sumber daya dalam latensi maksimum yang ditentukan. Tambahkan kebijakan tambahan untuk mengurangi latensi maksimum secara bertahap. Untuk menambahkan kebijakan tambahan, pilih Tambahkan kebijakan.
4. Di bawah lokasi penempatan sesi Game, pilih lokasi untuk disertakan dalam antrean. Secara default Semua lokasi disertakan. Semua armada dalam antrian harus memiliki konfigurasi sertifikat yang sama. Semua armada harus menjalankan build game yang kompatibel dengan klien game yang menggunakan antrian.
5. Di bawah Urutan tujuan, tambahkan satu atau beberapa tujuan ke antrean.
   a. Pilih Tambahkan tujuan.
   b. Pilih Lokasi tempat tujuan berada.
   c. Pilih jenis untuk tujuan Anda.
   d. Dari daftar nama armada atau alias yang dihasilkan, pilih salah satu nama yang ingin Anda tambahkan.
   e. Jika Anda memiliki beberapa tujuan, atur urutan default dengan menyeret ikon enam titik ke kiri tujuan. Amazon GameLift menggunakan urutan ini saat mencari tujuan untuk sumber daya yang tersedia untuk menempatkan sesi permainan baru.
6. Untuk prioritas penempatan sesi Game, tambahkan dan seret nilai Latensi, Biaya, Tujuan, dan Lokasi untuk menentukan cara Amazon GameLift memprioritaskan armada dalam antrean Anda. Untuk informasi lebih lanjut tentang memprioritaskan armada, lihat. Prioritaskan penempatan sesi game

7. Tambahkan lokasi ke urutan Lokasi Anda dan seret ke prioritas yang harus digunakan antrean. Jika Lokasi adalah prioritas terakhir untuk penempatan sesi game, Amazon GameLift menggunakannya sebagai tiebreaker.
8. (Opsional) Di bawah Pengaturan pemberitahuan acara lakukan hal berikut:
   a. Pilih atau buat topik SNS untuk menerima pemberitahuan peristiwa terkait penempatan. Untuk informasi selengkapnya tentang pemberitahuan acara, lihatAtur notifikasi kejadian untuk penempatan sesi game.
   b. Tambahkan data peristiwa khusus untuk ditambahkan ke acara yang dibuat oleh antrian ini.
9. (Opsional) Tambahkan Tag. Untuk informasi selengkapnya tentang pemberian tag, lihat Menandai sumber daya AWS.
10. Pilih Create (Buat).

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 lokasi di`us-east-1`,`us-east-2`. `us-west-2`, dan `ca-central-1`
- Memprioritaskan tujuan berdasarkan biaya dan kemudian lokasi dalam urutan yang ditentukan.

```
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="LOCATION","DESTINATION",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 nilai ARN armada dan alias dengan memanggil atau menggambarkan-alias dengan armada describe-fleet-attributesatau alias ID.

Jika `create-game-session-queue` permintaan berhasil, Amazon GameLift mengembalikan GameSessionQueueobjek dengan konfigurasi antrian baru. Anda sekarang dapat mengirimkan permintaan ke antrian menggunakan. StartGameSessionPlacement

Example Membuat antrean 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 mengembalikan GameSessionQueueobjek dengan konfigurasi antrian baru.

Atur notifikasi kejadian untuk penempatan sesi game

Anda dapat menggunakan notifikasi acara untuk memantau status permintaan penempatan individual. Sebaiknya siapkan notifikasi acara untuk semua game dengan aktivitas penempatan volume tinggi.

Terdapat dua pilihan untuk mengatur notifikasi kejadian.

- Minta Amazon GameLift mempublikasikan pemberitahuan acara ke topik Amazon Simple Notification Service (Amazon SNS) menggunakan antrean.
- Gunakan EventBridge peristiwa Amazon yang dipublikasikan secara otomatis dan rangkaian alatnya untuk mengelola acara.

Untuk daftar acara penempatan sesi game yang dipancarkan oleh AmazonGameLift, lihat. Acara penempatan sesi game

Menyiapkan topik SNS

GameLiftAgar Amazon dapat mempublikasikan semua peristiwa yang dihasilkan oleh antrean sesi game ke suatu topik, tetapkan bidang target notifikasi ke topik.

Untuk menyiapkan topik SNS untuk pemberitahuan GameLift peristiwa Amazon

1. Masuk ke AWS Management Console dan buka konsol Amazon SNS di https://console.aws.amazon.com/sns/v3/home.
2. Dari halaman Topik SNS, pilih Buat topik dan ikuti petunjuk untuk membuat topik Anda.
3. Berdasarkan 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 antrean selama pembuatan antrian, atau edit antrean yang ada untuk menambahkannya.

Mengatur 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 dalam AWS Key Management Service ()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 secara otomatis memposting semua acara penempatan sesi game keEventBridge. Dengan EventBridge Anda dapat mengatur aturan untuk memiliki peristiwa diarahkan ke target untuk diproses. Misalnya, Anda dapat menetapkan aturan untuk merutekan acara `PlacementFulfilled` ke AWS Lambda fungsi yang menangani tugas yang mendahului menghubungkan ke sesi permainan. Untuk informasi selengkapnya tentang EventBridge, Apa itu Amazon EventBridge? dalam Panduan Pengguna Amazon EventBridge.

Berikut ini adalah beberapa contoh EventBridge aturan yang digunakan dengan GameLift antrian Amazon:

Pertandingan acara dari semua GameLift antrian Amazon

```
{
    "source": [
        "aws.gamelift"
    ],
    "detail-type": [
        "GameLift Queue Placement Event"
    ]
}
```

Mencocokkan acara dari antrean tertentu

```
{
    "source": [
        "aws.gamelift"
    ],
    "detail-type": [
        "GameLift Queue Placement Event"
    ],
    "resources": [
        "arn:aws:gamelift:your_region:your_account:gamesessionqueue/your_queue_name"
    ]
}
```

Tutorial: Mengatur antrean sesi game untuk Instans Spot

Pengantar

Tutorial ini menjelaskan cara mengatur penempatan sesi game untuk game yang digunakan pada armada Spot berbiaya rendah. Armada spot memerlukan langkah-langkah tambahan untuk mempertahankan ketersediaan server game terus-menerus untuk pemain Anda.

Audiens yang dituju

Tutorial ini ditujukan untuk pengembang game yang ingin menggunakan armada Spot untuk meng-host server game khusus atau Server Realtime.

Apa yang akan Anda pelajari

- Tentukan kelompok pemain yang melayani 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 antrean, tambahkan armada, dan prioritaskan tempat Amazon menempatkan sesi game. GameLift
- Tambahkan kebijakan latensi pemain untuk membantu meminimalkan masalah latensi.

Prasyarat

Sebelum membuat armada dan antrian untuk penempatan sesi game, selesaikan tugas-tugas berikut:

- UlasanCara GameLift kerja Amazon.
- Integrasikan server game Anda dengan Amazon GameLift.
- Unggah build server game atau skrip Realtime Anda ke AmazonGameLift.
- 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 bukanlah 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.

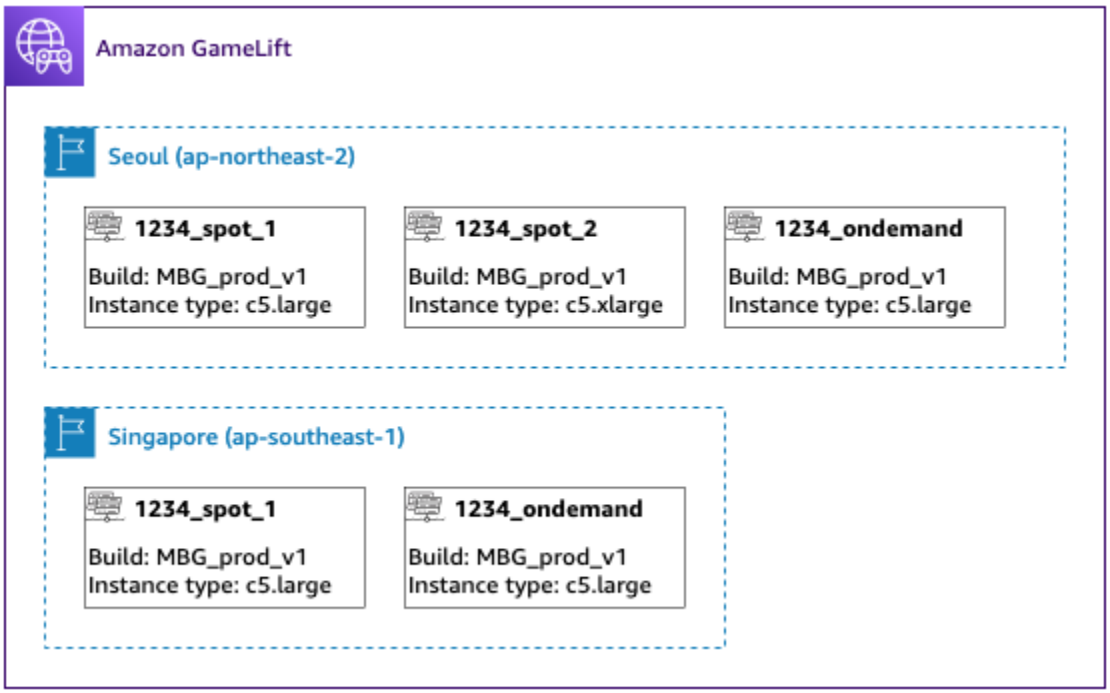
Untuk informasi selengkapnya, lihat Tentukan cakupan antrean Anda.

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-timur laut-2 (Seoul) dan 2 armada di lokasi ap-tenggara 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 instans c5.large, armada 1234_spot_2 dengan tipe instance c5.xlarge, dan armada 1234_ondemand dengan tipe instance c5.large. Armada di ap-tenggara 1 berisi konfigurasi armada berikut: armada 1234_spot_1 dengan tipe instans c5.large dan armada 1234_ondemand dengan tipe instance c5.large.

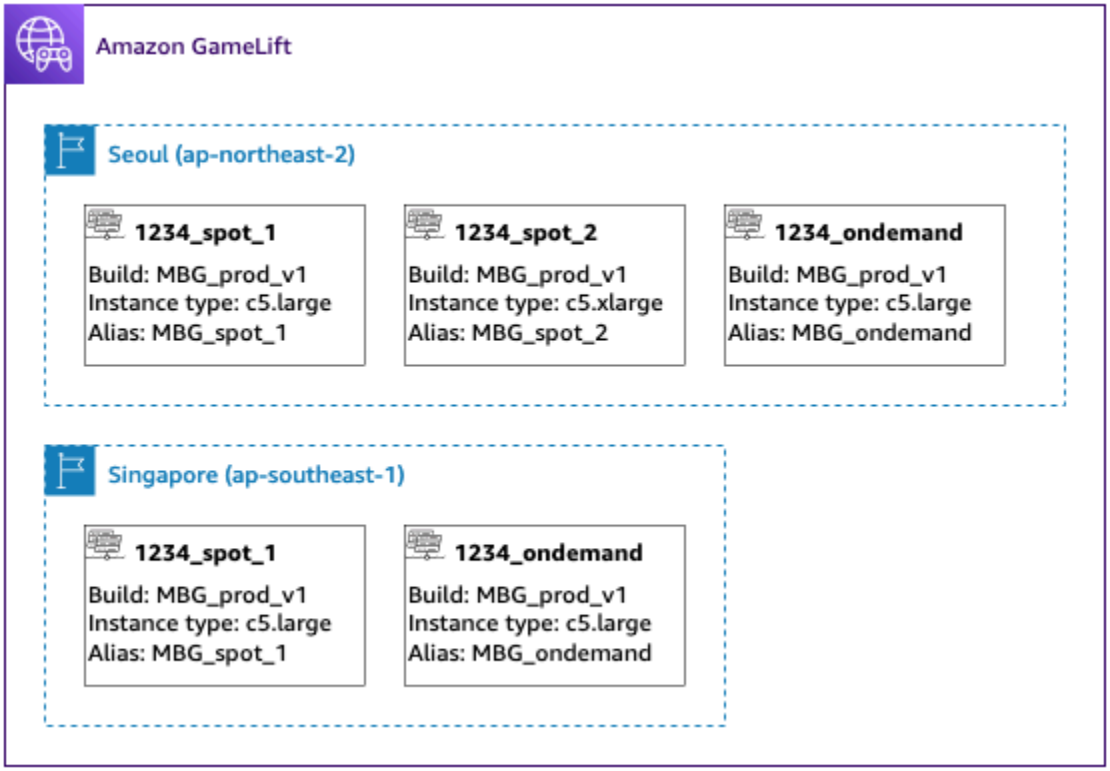


Langkah 3: Tetapkan alias untuk setiap armada

Buat alias baru untuk setiap armada dalam infrastruktur Anda. Alias identitas armada abstrak, membuat penggantian armada periodik efisien. Untuk informasi selengkapnya tentang membuat alias, lihatTambahkan alias ke armada Amazon GameLift .

Infrastruktur armada kami memiliki lima armada, jadi kami membuat lima alias menggunakan strategi routing. Kita 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 Membuat antrean 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

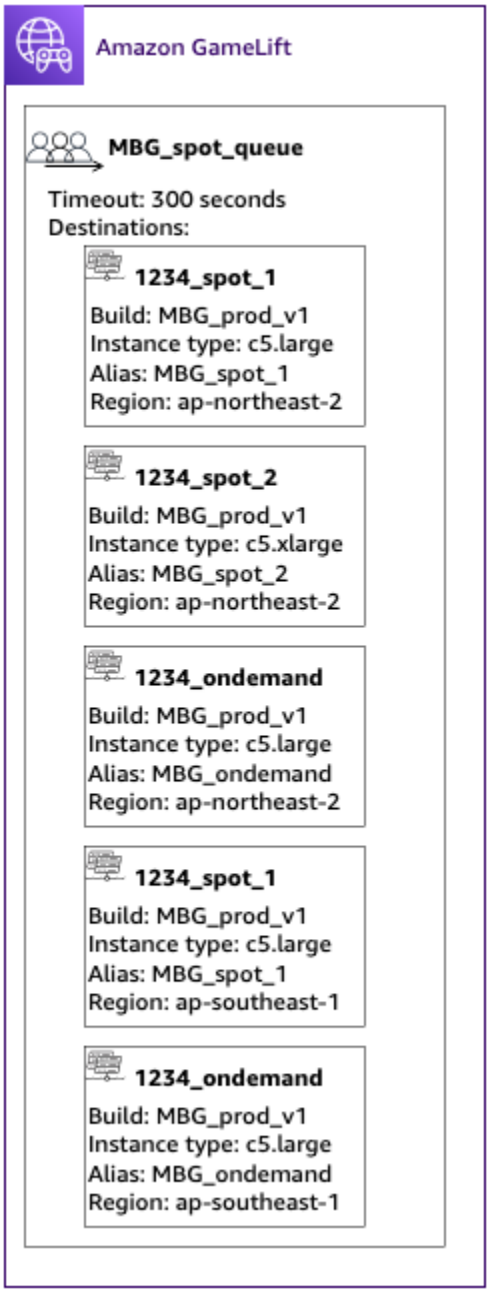
Saat membuat antrean Anda:

- Atur batas waktu default menjadi 10 menit. Nantinya, Anda dapat menguji bagaimana batas waktu antrian memengaruhi waktu tunggu pemain Anda untuk masuk ke game.
- 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 antrean.

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 berupaya 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 menempatkan sesi permainan di armada Singapura.

Diagram berikut menunjukkan antrian dengan batas waktu 300 detik dan tujuan yang diprioritaskan. Tujuannya 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_spot_1 di ap-tenggara - 1, dan 1234_ondemand di ap-tenggara 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 menciptakan sesi permainan untuk sekelompok pemain. Kita bisa meminta pemain menunggu sedikit lebih lama untuk masuk ke game dengan pengalaman gameplay yang ideal. Tes permainan kami menunjukkan pengamatan berikut:

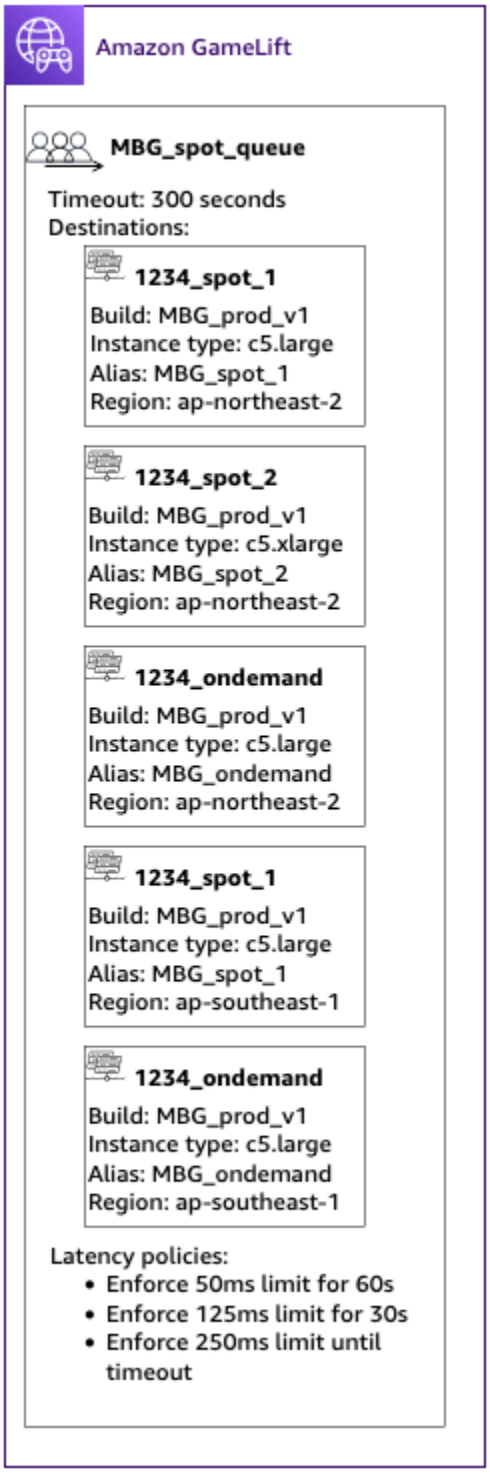
- Latensi di bawah 50 milidetik sangat ideal.

- 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 diijinkan. 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 keempat dengan kebijakan latensi pemain ditambahkan. Kebijakan latensi pemain menyatakan, menegakkan batas 50ms selama 60-an, menegakkan batas 125 ms selama 30-an, dan menegakkan batas 250ms hingga batas waktu tunggu.



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 permainan untuk pemain ini, rujuk nama antrian sesi permainan ini dalam permintaan. Untuk informasi selengkapnya tentang membuat permintaan penempatan sesi game, lihatBuat sesi permainan, atauMengintegrasikan klien game untuk Server Realtime.

Langkah selanjutnya:

- Rancang antrian Anda sendiri.
- Buat antrian.
- Gunakan antrian dengan klien game Anda.

Mengelola sumber daya menggunakan AWS CloudFormation

Anda dapat menggunakan AWS CloudFormation untuk mengelola GameLift sumber daya Amazon Anda. Masuk AWS CloudFormation, Anda membuat templat yang mendasari setiap sumber daya dan kemudian menggunakan templat tersebut untuk membuat sumber daya Anda. Untuk memperbarui sumber daya, Anda membuat perubahan pada templat Anda dan menggunakan AWS CloudFormation untuk menerapkan pembaruan. Anda dapat mengatur sumber daya Anda ke dalam grup logika, yang disebut tumpukan dan set tumpukan.

Menggunakan AWS CloudFormation untuk mempertahankan sumber daya GameLift hosting Amazon Anda menawarkan cara yang lebih efisien untuk mengelola kumpulan AWS sumber 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 tentang AWS CloudFormation, lihat AWS CloudFormationPanduan Pengguna . Untuk melihat informasi template untuk GameLift sumber daya Amazon, lihat referensi jenis GameLift sumber daya Amazon.

Praktik terbaik

Untuk panduan terperinci tentang penggunaanAWS CloudFormation, lihat praktik AWS CloudFormation terbaik dalam Panduan AWS CloudFormation Pengguna. Selain itu, praktik terbaik ini memiliki relevansi khusus dengan AmazonGameLift.

- Kelola sumber daya Anda secara konsisten. AWS CloudFormation Jika Anda mengubah sumber daya Anda di luar sumber daya AWS CloudFormation Anda akan keluar dari sinkron dengan template 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 Anda untuk mengganti build, AWS CloudFormation gantikan armada yang terhubung ke build. AWS CloudFormationkemudian memperbarui alias yang ada untuk menunjuk ke armada baru. Untuk informasi selengkapnya, lihat Bekerja dengan tumpukan di Panduan AWS CloudFormation Pengguna.
  - Gunakan set tumpukan AWS CloudFormation jika Anda men-deploy tumpukan identik di beberapa wilayah atau akun AWS. Untuk informasi selengkapnya, lihat Bekerja dengan set tumpukan di AWS CloudFormationUser Guide.
- 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 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 antrean sesi game yang digunakannya.
- Buat alias terpisah untuk setiap armada di tumpukan.

Menggunakan AWS CloudFormation tumpukan

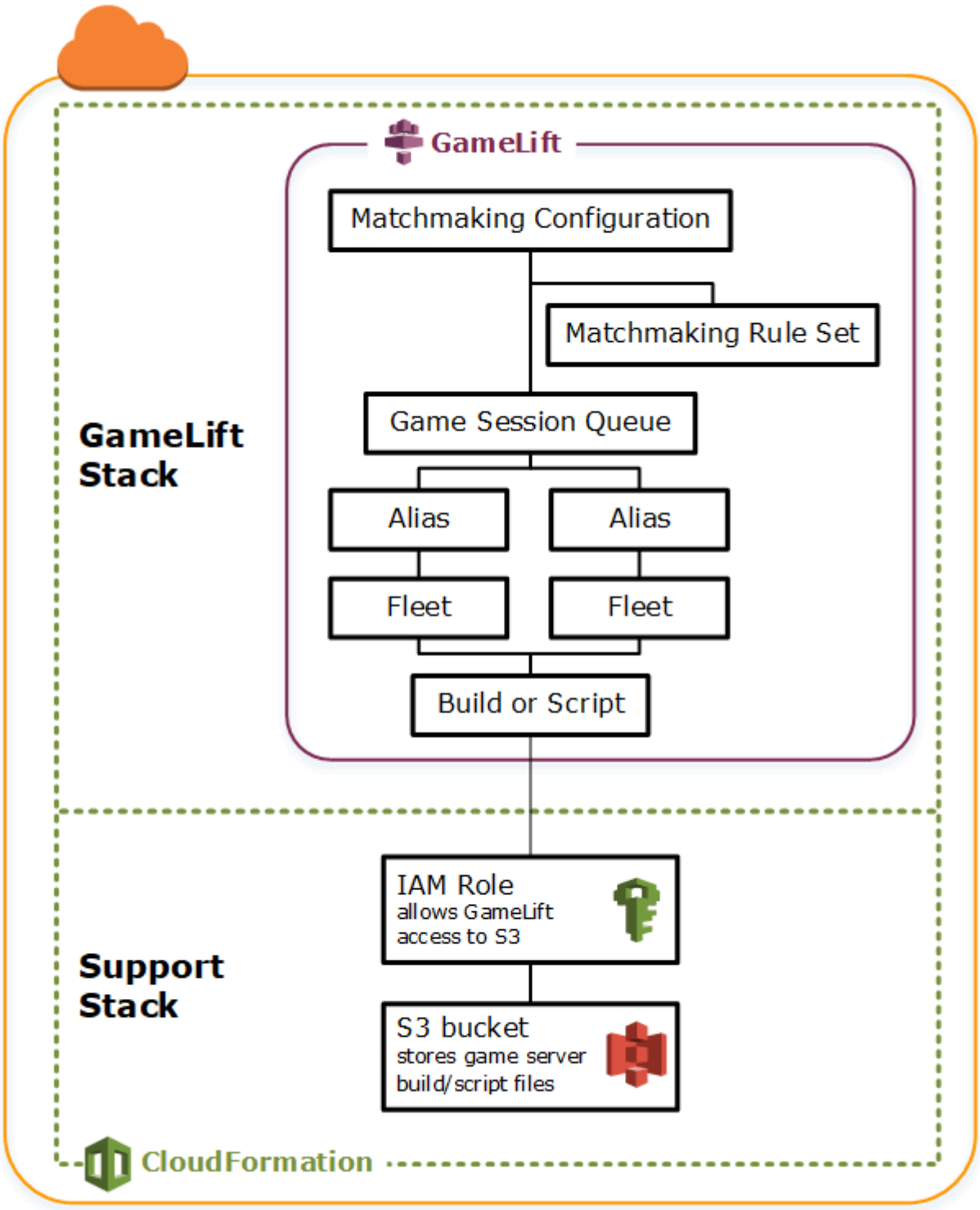
Kami merekomendasikan struktur berikut untuk digunakan saat menyiapkan AWS CloudFormation tumpukan untuk GameLift sumber daya Amazon. Struktur tumpukan optimal Anda bervariasi tergantung pada apakah Anda menerapkan game Anda di satu lokasi atau beberapa lokasi.

Tumpukan untuk satu lokasi

Untuk mengelola GameLift sumber daya Amazon di satu lokasi, kami merekomendasikan struktur dua tumpukan:

- Tumpukan dukungan - Tumpukan ini berisi sumber daya yang bergantung pada GameLift sumber daya Amazon Anda. 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 GameLift izin Amazon untuk mengambil file Anda dari bucket S3 saat membuat sumber daya GameLift build atau skrip Amazon. Tumpukan ini mungkin juga berisi sumber daya AWS yang digunakan dengan game Anda, seperti tabel DynamoDB, klaster Amazon Redshift, dan fungsi Lambda.
- Amazon GameLift stack - Tumpukan ini berisi semua GameLift sumber daya Amazon Anda, termasuk build atau skrip, satu set armada, alias, dan antrean sesi game. AWS CloudFormationmembuat sumber daya build atau skrip dengan file yang disimpan di lokasi bucket S3 dan menyebarkan build atau skrip ke satu atau lebih 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 menggambarkan struktur dua tumpukan untuk men-deploy sumber daya dalam satu Wilayah AWS.



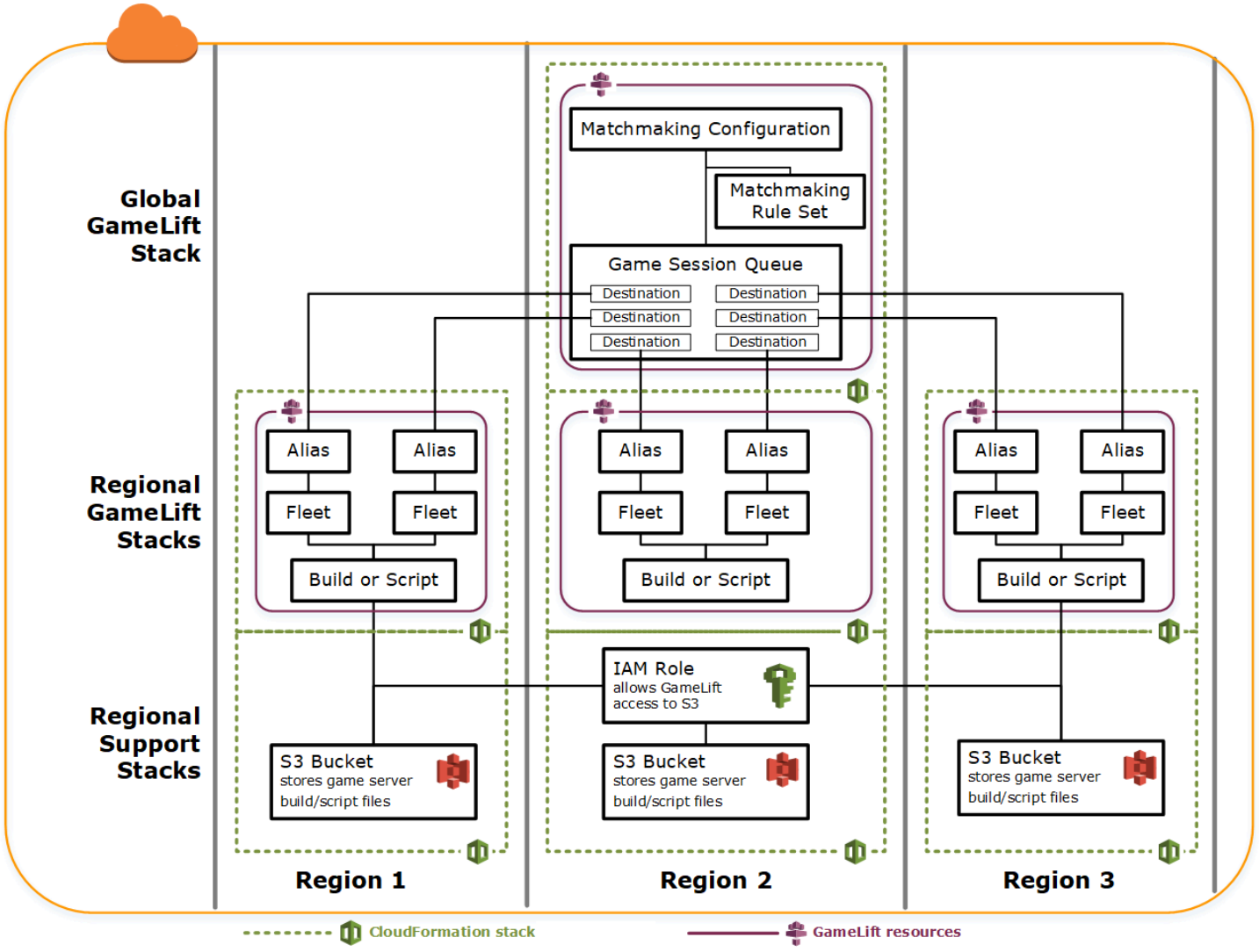
Tumpukan untuk beberapa wilayah

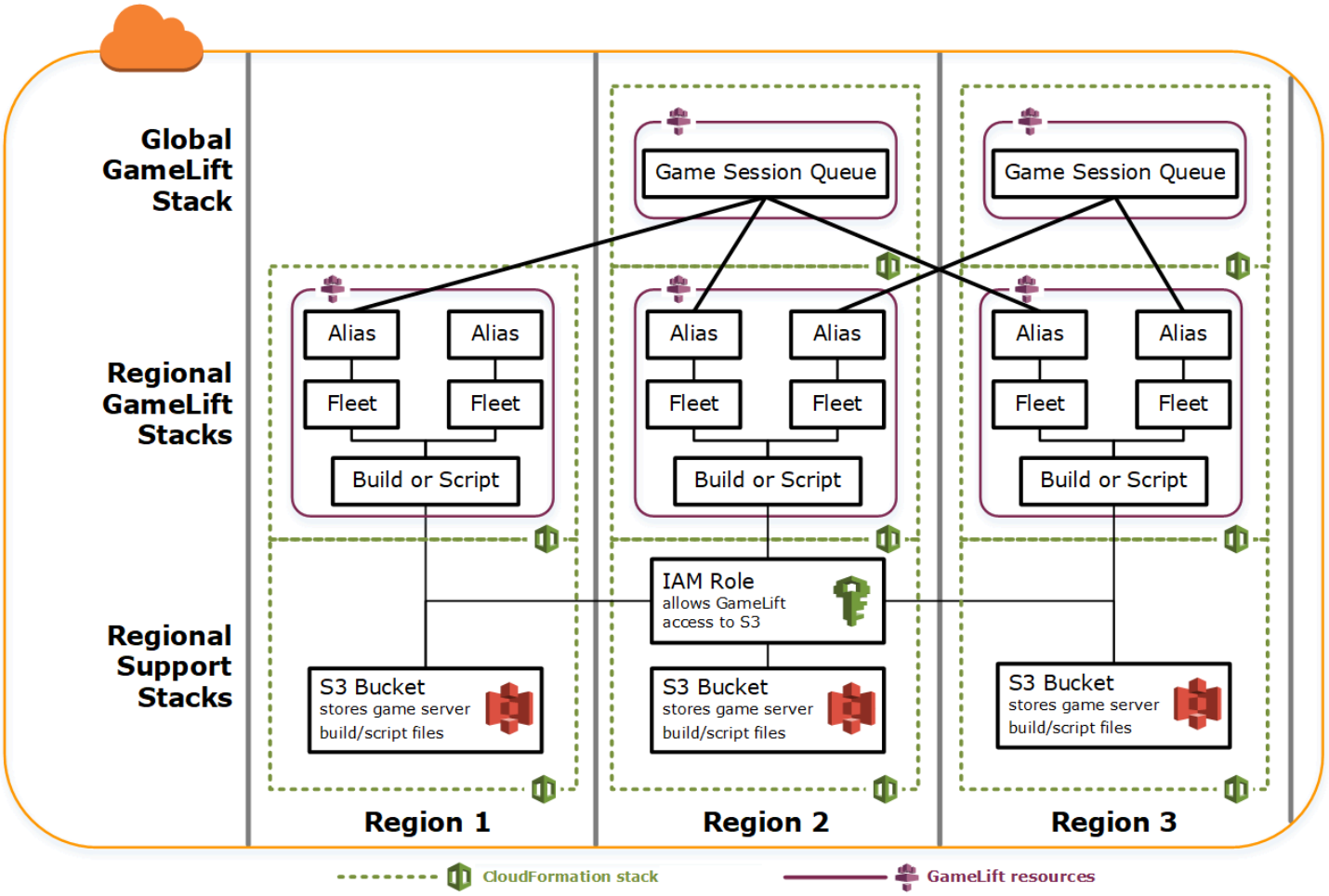
Ketika men-deploy game Anda di lebih dari satu Wilayah, perlu diingat bagaimana sumber daya dapat berinteraksi di seluruh Wilayah. Beberapa sumber daya, seperti GameLift armada Amazon, hanya dapat mereferensikan sumber daya lain di Wilayah yang sama. Sumber daya lain, seperti

GameLift antrian Amazon, adalah Region agnostik. Untuk mengelola GameLift sumber daya Amazon di beberapa Wilayah, kami merekomendasikan struktur berikut.

- Tumpukan dukungan regional — Tumpukan ini berisi sumber daya yang bergantung pada sumber GameLift daya Amazon Anda. Tumpukan ini harus mencakup bucket S3 di mana Anda menyimpan server game kustom atau file skrip Realtime. Ini mungkin juga berisi sumber daya AWS untuk game Anda, seperti tabel DynamoDB, klaster Amazon Redshift, dan fungsi Lambda. Banyak dari sumber daya ini adalah khusus Wilayah, sehingga Anda harus membuat sumber daya di setiap Wilayah. Amazon GameLift juga memerlukan 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.
- GameLiftTumpukan Amazon Regional —Tumpukan ini berisi GameLift sumber daya Amazon yang harus ada di setiap wilayah tempat game Anda digunakan, termasuk build atau skrip, satu set armada, dan alias. AWS CloudFormationmembuat sumber daya build atau skrip dengan file di lokasi bucket S3, dan menyebarkan build atau skrip ke satu atau lebih 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 stack - Tumpukan ini berisi antrean sesi game dan sumber daya perjodohan 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 menggambarkan struktur dua tumpukan untuk men-deploy sumber daya dalam beberapa Wilayah AWS. Diagram pertama menunjukkan struktur untuk antrean sesi game tunggal. Diagram kedua menunjukkan struktur dengan beberapa antrean.


Global GameLift Stack
Regional GameLift Stacks
Regional Support Stacks
Matchmaking Configuration
Matchmaking Rule Set
Game Session Queue
Destination
Alias
Fleet
Build or Script
IAM Role
allows GameLift access to S3
S3 Bucket
stores game server build/script files
Region 1
Region 2
Region 3
CloudFormation stack
GameLift resources



Memperbarui build

Amazon GameLift membangun tidak berubah, seperti hubungan antara membangun 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 berisi sumber daya build, armada, dan alias terkait, perilaku default AWS CloudFormation adalah secara otomatis melakukan langkah-langkah ini secara berurutan. Anda memicu pembaruan ini dengan terlebih dahulu mengunggah file build baru ke lokasi S3 baru. Kemudian Anda memodifikasi templat build AWS CloudFormation 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 build Amazon baru. GameLift
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 memiliki sumber daya AWS CloudFormation dipertahankan alih-alih menghapusnya. Untuk informasi selengkapnya, lihat RetainResources dalam Referensi API AWS CloudFormation.

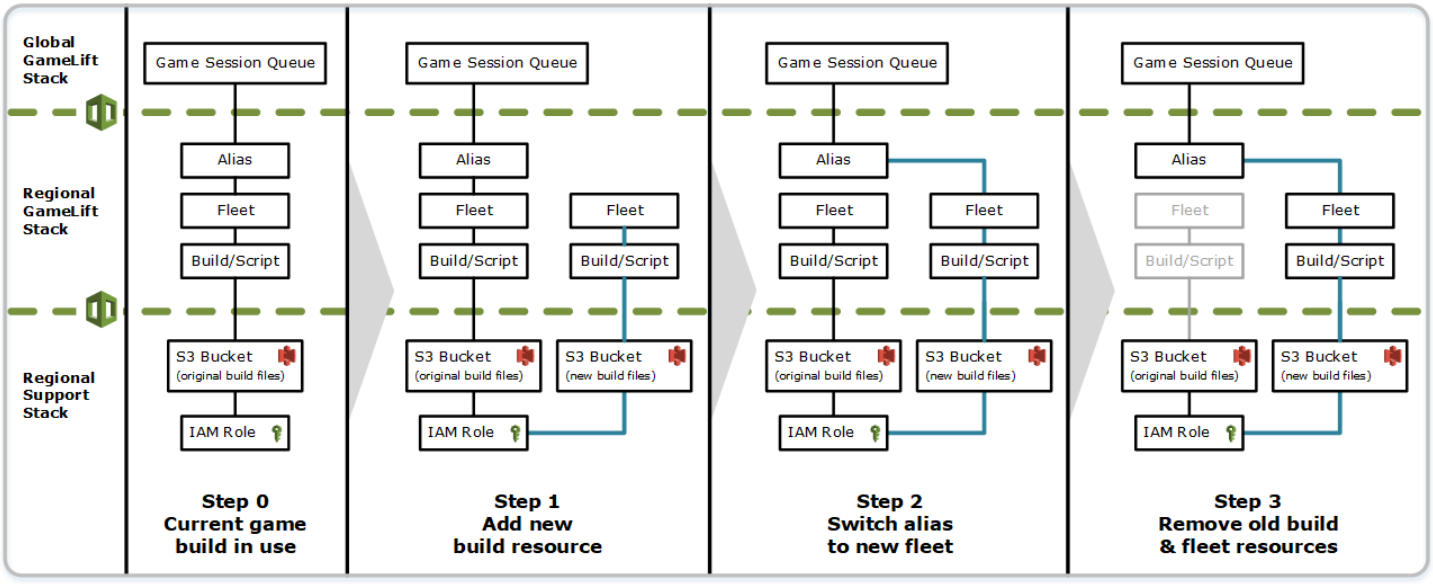
Menyebarkan pembaruan build secara manual

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 GameLift konsol Amazon 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 templat, AWS CloudFormation menciptakan 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. GameLiftPembuatan dan armada Amazon baru 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 CloudFormationPanduan Pengguna.

VPC mengintip untuk Amazon GameLift

Topik ini memberikan panduan tentang cara mengatur koneksi peering VPC antara server game yang GameLift di-host Amazon Anda dan sumber daya non-Amazon lainnya. GameLift Gunakan koneksi mengintip Amazon Virtual Private Cloud (VPC) untuk memungkinkan server game Anda berkomunikasi secara langsung dan privat dengan sumber dayaAWS, seperti layanan web atau

repositori. Anda dapat membuat peering VPC dengan sumber daya apa pun yang berjalan di AWS dan dikelola oleh akun AWS yang dapat Anda akses.

> **Note**
>
> Peering VPC adalah fitur lanjutan. Untuk mempelajari tentang opsi pilihan untuk mengaktifkan server game Anda agar dapat berkomunikasi secara langsung dan privat dengan sumber daya AWS Anda yang lain, lihat Berkomunikasi dengan sumber daya AWS lain dari armada.

Jika Anda sudah terbiasa dengan VPC Amazon dan VPC peering, pahami bahwa mengatur peering dengan server GameLift game Amazon agak berbeda. Anda tidak memiliki akses ke VPC yang berisi server game Anda—itu dikendalikan oleh GameLift layanan Amazon—sehingga Anda tidak dapat langsung meminta VPC peering untuk itu. Sebagai gantinya, pertama-tama Anda melakukan pra-otorisasi VPC dengan GameLift sumber daya non-Amazon Anda untuk menerima permintaan peering dari layanan Amazon. GameLift Kemudian Anda memicu Amazon GameLift untuk meminta VPC mengintip yang baru saja Anda otorisasi. Amazon GameLift menangani tugas membuat koneksi peering, menyiapkan tabel rute, dan mengonfigurasi koneksi.

Menyiapkan peering VPC untuk armada yang sudah ada

1. Dapatkan ID AWS akun dan kredensi.

   Anda memerlukan ID dan kredensial masuk untuk akun AWS. Anda dapat menemukan ID akun AWS dengan masuk ke AWS Management Console dan melihat pengaturan akun Anda. Untuk mendapatkan kredensial, buka konsol IAM.

   - AWSakun yang Anda gunakan untuk mengelola server GameLift game Amazon Anda.
   - AWSakun yang Anda gunakan untuk mengelola GameLift sumber daya non-Amazon Anda.

   Jika Anda menggunakan akun yang sama untuk GameLift sumber daya Amazon GameLift dan non-Amazon, Anda hanya memerlukan ID dan kredensi untuk akun tersebut.

2. Dapatkan pengenal untuk setiap VPC.

   Dapatkan informasi berikut untuk kedua VPC yang akan peered:

   - VPC untuk server GameLift game Amazon Anda - Ini adalah ID GameLift armada Amazon Anda. Server game Anda diterapkan di Amazon GameLift pada armada instans EC2. Armada

secara otomatis ditempatkan di VPC-nya sendiri, yang dikelola oleh layanan AmazonGameLift. Anda tidak memiliki akses langsung ke VPC, sehingga diidentifikasi oleh ID armada.

- VPC untuk GameLift AWS sumber daya non-Amazon Anda — Anda dapat membuat VPC mengintip 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 dengan Amazon VPC. Setelah membuat VPC, Anda dapat menemukan ID VPC dengan masuk ke AWS Management Console untuk Amazon VPC dan melihat VPC anda.

> **Note**
>
> Saat menyiapkan peering, kedua VPC harus ada di wilayah yang sama. VPC untuk server game GameLift armada Amazon Anda berada di wilayah yang sama dengan armada.

3. Mengotorisasi peering VPC.

Pada langkah ini, Anda melakukan pra-otorisasi permintaan di masa mendatang dari Amazon GameLift untuk mengintip VPC dengan server game Anda dengan VPC Anda untuk sumber daya non-Amazon. GameLift Tindakan ini memperbarui grup keamanan untuk VPC Anda.

Untuk mengotorisasi peering VPC, panggil Amazon GameLift service API CreateVpcPeeringAuthorization() atau gunakan perintah CLI. AWS `create-vpc-peering-authorization` Lakukan panggilan ini menggunakan akun yang mengelola sumber daya non-Amazon GameLift Anda. Tentukan informasi berikut:

- Peer VPC ID - Ini untuk VPC dengan sumber daya GameLift non-Amazon Anda.
- ID GameLift AWS akun Amazon - Ini adalah akun yang Anda gunakan untuk mengelola GameLift armada Amazon Anda.

Setelah Anda mengotorisasi peering VPC, otorisasi tetap berlaku selama 24 jam kecuali dicabut. Anda dapat mengelola otorisasi peering VPC menggunakan operasi berikut:

- DescribeVpcPeeringAuthorizations() (AWSCLI`describe-vpc-peering-authorizations`).
- DeleteVpcPeeringAuthorization() (AWSCLI`delete-vpc-peering-authorization`).

4. Meminta koneksi peering.

   Dengan otorisasi yang valid, Anda dapat meminta agar Amazon GameLift membuat koneksi peering.

   Untuk meminta peering VPC, panggil Amazon GameLift service API CreateVpcPeeringConnection() atau gunakan perintah AWS CLI. `create-vpc-peering-connection` Lakukan panggilan ini menggunakan akun yang mengelola server GameLift game Amazon Anda. Gunakan informasi berikut untuk mengidentifikasi VPC dua yang ingin Anda peering:

   - ID VPC rekan dan ID AWS akun — Ini adalah VPC untuk GameLift sumber daya non-Amazon Anda dan akun yang Anda gunakan untuk mengelolanya. ID VPC harus sesuai dengan ID pada otorisasi peering yang valid.
   - Fleet ID — Ini mengidentifikasi VPC untuk server game Amazon GameLift Anda.

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 sukses dan kegagalan.

Setelah koneksi peering dibuat, Anda dapat mengelolanya menggunakan operasi berikut:

- DescribeVpcPeeringConnections() (AWSCLI`describe-vpc-peering-connections`).
- DeleteVpcPeeringConnection() (AWSCLI`delete-vpc-peering-connection`).

Mengatur peering VPC dengan armada baru

Anda dapat membuat GameLift armada Amazon baru dan meminta koneksi peering VPC secara bersamaan.

1. Dapatkan ID AWS akun dan kredensi.

   Anda memerlukan ID dan kredensial masuk untuk dua akun AWS berikut. Anda dapat menemukan ID akun AWS dengan masuk ke AWS Management Console dan melihat pengaturan akun Anda. Untuk mendapatkan kredensial, buka konsol IAM.

   - AWSakun yang Anda gunakan untuk mengelola server GameLift game Amazon Anda.
   - AWSakun yang Anda gunakan untuk mengelola GameLift sumber daya non-Amazon Anda.

   Jika Anda menggunakan akun yang sama untuk GameLift sumber daya Amazon GameLift dan non-Amazon, Anda hanya memerlukan ID dan kredensi untuk akun tersebut.

2. Dapatkan ID VPC untuk sumber daya non-Amazon GameLift AWS Anda.

   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 GameLift sumber daya non-Amazon Anda dikelola di bawah AWS akun atau grup pengguna/pengguna yang berbeda dari yang Anda gunakan dengan AmazonGameLift, Anda harus menggunakan kredensi akun ini saat meminta otorisasi di langkah berikutnya.

   Setelah membuat VPC, Anda dapat menemukan ID VPC di konsol Amazon VPC dengan melihat VPC Anda.

3. Otorisasi peering VPC dengan sumber daya non-Amazon. GameLift

   Saat Amazon GameLift membuat armada baru dan VPC yang sesuai, Amazon juga mengirimkan permintaan untuk mengintip dengan VPC untuk sumber daya non-Amazon Anda. GameLift Anda harus melakukan pra-otorisasi permintaan tersebut. Langkah ini memperbarui grup keamanan untuk VPC Anda.

   Menggunakan kredensi akun yang mengelola GameLift sumber daya non-Amazon Anda, hubungi Amazon GameLift service API CreateVpcPeeringAuthorization() atau gunakan perintah CLI. AWS `create-vpc-peering-authorization` Tentukan informasi berikut:

   - Peer VPC ID — ID VPC dengan sumber daya GameLift non-Amazon Anda.
   - ID GameLift AWS akun Amazon — ID akun yang Anda gunakan untuk mengelola GameLift armada Amazon Anda.

Setelah Anda mengotorisasi peering VPC, otorisasi tetap berlaku selama 24 jam kecuali dicabut. Anda dapat mengelola otorisasi peering VPC menggunakan operasi berikut:

- DescribeVpcPeeringAuthorizations() (AWSCLI`describe-vpc-peering-authorizations`).
- DeleteVpcPeeringAuthorization() (AWSCLI`delete-vpc-peering-authorization`).

4. Ikuti instruksi untuk membuat armada baru menggunakan CLI AWS. Sertakan parameter tambahan berikut:

   - peer-vpc-aws-account-id — ID untuk akun yang Anda gunakan untuk mengelola VPC dengan sumber daya GameLift non-Amazon Anda.
   - peer-vpc-id— ID VPC dengan GameLift akun non Anda.

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 ID GameLift AWS akun non-Amazon Anda 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 dalam membuat koneksi peering VPC untuk server GameLift game Amazon Anda, pertimbangkan akar penyebab umum ini:

- Otorisasi untuk koneksi yang diminta tidak ditemukan:

  - Periksa status otorisasi VPC untuk VPC GameLift non-Amazon. Mungkin tidak ada atau mungkin sudah kedaluwarsa.
  - Periksa wilayah dari dua VPC yang ingin Anda coba peer. Jika mereka tidak berada di wilayah yang sama, mereka tidak dapat di-peer.
- Blok CIDR (lihat Konfigurasi koneksi peering VPC tidak valid) dari dua VPC Anda tumpang tindih. Blok IPv4 CIDR yang ditugaskan untuk melakukan peering VPC tidak boleh tumpang tindih. Blok CIDR VPC untuk GameLift armada Amazon Anda secara otomatis ditetapkan dan tidak dapat diubah, jadi Anda harus mengubah blok CIDR untuk VPC untuk sumber daya non-Amazon Anda. GameLift Untuk menyelesaikan masalah ini:
  - Cari blok CIDR ini untuk GameLift armada Amazon Anda dengan menelepon. `DescribeVpcPeeringConnections()`
  - Buka konsol Amazon VPC, temukan VPC untuk GameLift sumber daya non-Amazon Anda, dan ubah blok CIDR agar 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.

Melihat data game Anda di konsol

GameLiftLayanan Amazon yang dikelola terus mengumpulkan data untuk game aktif untuk membantu Anda memahami perilaku dan kinerja pemain. Dengan GameLift konsol Amazon, Anda dapat melihat, mengelola, dan menganalisis informasi ini untuk build, armada, sesi game, dan sesi pemain Anda.

Topik

Melihat GameLift status Amazon Anda saat ini

GameLiftDasbor Amazon menyediakan tampilan berikut:

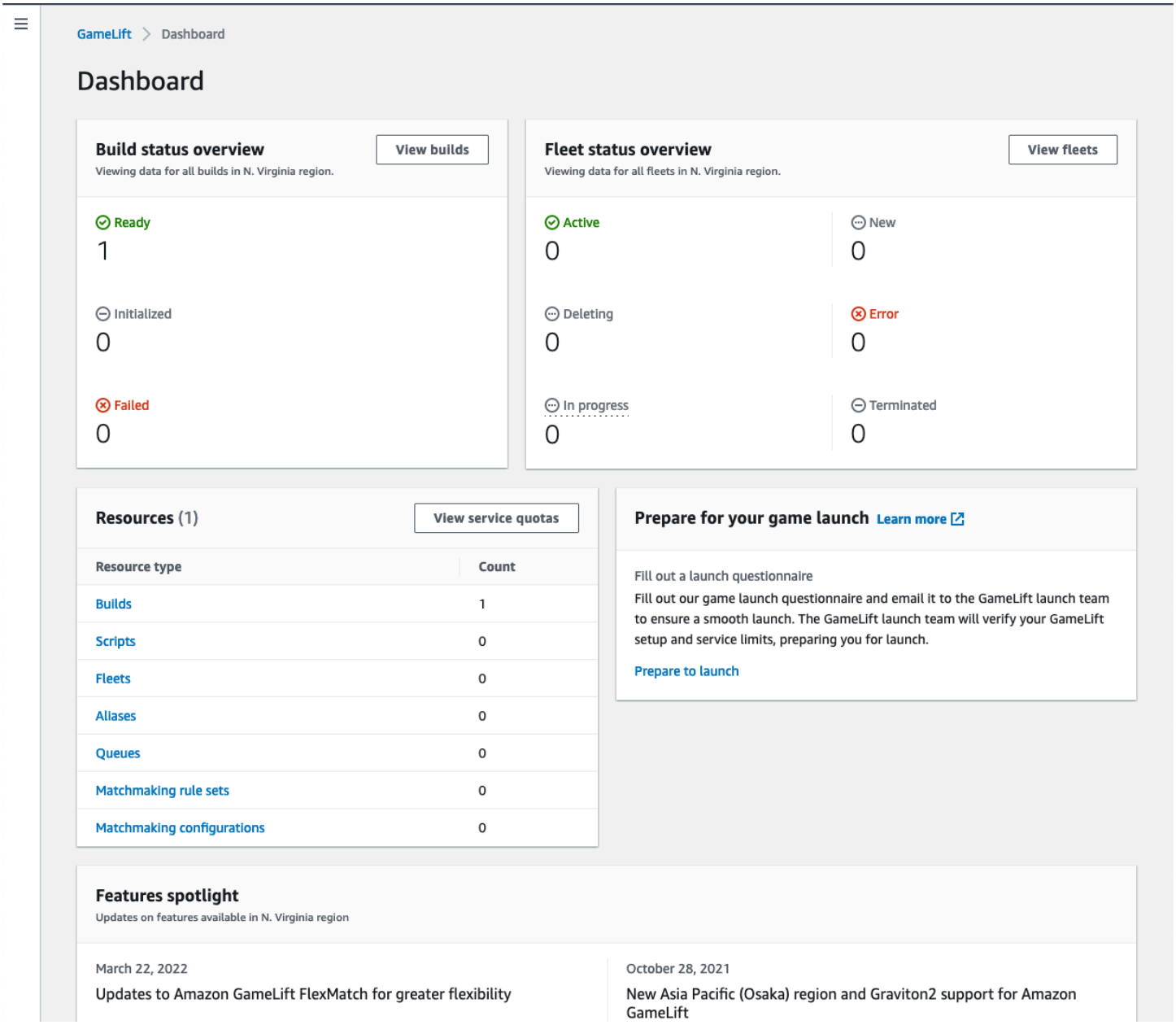
- Jumlah build dalam status Ready, Initialized, dan Failed. Pilih Lihat build untuk detail tentang build di Wilayah Anda saat ini.
- Jumlah armada dalam semua status. Pilih Lihat armada untuk detail tentang armada di Wilayah Anda saat ini.
- Sumber daya Anda saat ini.
- Kesehatan keseluruhan layanan Anda, menurutAWS Health. Untuk informasi selengkapnya, lihat Konsep untuk AWS Health di Panduan AWS Health Pengguna.
- Pengumuman fitur dan layanan baru.

Untuk membuka GameLift dasbor Amazon

- Di GameLiftkonsol Amazon, di panel navigasi, pilih Dasbor.

Dari dasbor, Anda dapat:

- Siapkan permainan Anda untuk diluncurkan dengan memilih Siapkan peluncuran dan mengisi kuesioner peluncuran yang sesuai.
- Permintaan kuota layanan meningkat dalam persiapan peluncuran atau menanggapi peluncuran dengan memilih Lihat kuota layanan.
- Lihat posting blog dan informasi rinci tentang fitur baru dengan memilih tautan di sorotan Fitur.



Lihat build Anda

Di halaman Builds GameLift konsol Amazon, Anda dapat melihat informasi tentang dan mengelola semua build server game yang telah Anda unggah ke Amazon. GameLift Di panel navigasi, pilih Hosting, Build.

Halaman Builds menampilkan informasi berikut untuk setiap build:

> **Note**
>
> Halaman Build menampilkan build di AWS Wilayah Anda saat ini saja.

- Nama - Nama yang terkait dengan build yang diunggah.
- Status - Status membangun. Menampilkan salah satu dari tiga pesan status:
  - Diinisialisasi - Upload belum dimulai atau masih berlangsung.
  - Siap - Build siap untuk pembuatan armada.
  - Gagal — Build habis sebelum Amazon GameLift menerima biner.
- Waktu pembuatan - Tanggal dan waktu Anda mengunggah build ke AmazonGameLift.
- Build ID - ID unik yang ditetapkan untuk build on upload.
- Versi - Label versi yang terkait dengan build yang diunggah.
- Sistem operasi - OS tempat build berjalan. Build OS menentukan sistem operasi yang GameLift diinstal Amazon pada instans armada.
- Ukuran — Ukuran, dalam megabyte (MB), file build yang diunggah ke Amazon. GameLift
- Armada - Jumlah armada yang dikerahkan dengan membangun.

Dari halaman ini Anda dapat melakukan hal-hal berikut:

- Lihat detail bangunan. Pilih nama build untuk membuka halaman detail build.
- Buat armada baru dari sebuah bangunan. Pilih build, lalu pilih Create fleet.
- Filter dan urutkan daftar bangunan. Gunakan kontrol di bagian atas tabel.
- Menghapus bangunan. Pilih build, lalu pilih Hapus.

Membangun rincian

Pada halaman Builds, pilih nama build untuk membuka halaman detailnya. Bagian Ikhtisar pada halaman detail menampilkan informasi ringkasan build yang sama dengan halaman Build. Bagian Armada menunjukkan daftar armada yang dibuat dengan build, termasuk informasi ringkasan yang sama dengan halaman Armada.

Melihat skrip Anda

Pada halaman Skrip GameLift konsol Amazon, Anda dapat melihat informasi tentang dan mengelola semua skrip Server Realtime yang telah Anda unggah ke AmazonGameLift. Di panel navigasi, pilih Hosting, Skrip.

Halaman Script menampilkan informasi berikut untuk setiap skrip:

> **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 ke skrip saat diunggah.
- Versi - Label versi yang terkait dengan skrip upload.
- Ukuran — Ukuran, dalam megabyte (MB), dari file skrip yang diunggah ke Amazon. GameLift
- Waktu pembuatan - Tanggal dan waktu Anda mengunggah skrip ke AmazonGameLift.
- Armada - Jumlah armada yang dikerahkan dengan naskah.

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.

Detail 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.

Lihat armada Anda

Anda dapat melihat informasi tentang semua armada yang dibuat untuk meng-host game Anda di Amazon di GameLift bawah AWS akun Anda. Daftar menunjukkan armada yang dibuat 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 game, 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 informasi yang ditunjukkan dengan memilih tombol Pengaturan (roda gigi).

- Nama - Nama ramah 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 bisa berupa armada EC2 Terkelola atau armada Anywhere.
- Jenis instans — Jenis instans Amazon EC2, yang menentukan kapasitas komputasi instans armada.
- Instans aktif – Jumlah Instans EC2 yang digunakan untuk armada.
- Instans yang diinginkan - Jumlah instans EC2 untuk tetap aktif.
- Sesi permainan - Jumlah sesi permainan aktif yang berjalan di armada. Data tertunda lima menit.

Meninjau detail armada

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 ke armada ini. ARN armada mengidentifikasinya sebagai GameLift sumber daya Amazon dan menentukan wilayah dan akun. AWS
- Deskripsi - Deskripsi singkat tentang armada.
- Status – Status armada saat ini, yang mungkin saja Baru, Mengunduh, Membangun, dan Aktif.
- Waktu pembuatan - Tanggal dan waktu ketika armada diciptakan.
- Waktu penghentian - Tanggal dan waktu armada dihentikan. Ini kosong jika armada masih aktif.
- Tipe Armada – Menunjukkan apakah armada menggunakan sesuai permintaan atau instans spot.
- Tipe EC2 – Tipe Instans Amazon EC2 dipilih untuk armada saat diciptakan.
- Peran instans - Peran AWS IAM yang mengelola akses ke AWS sumber daya Anda yang lain, jika ada yang disediakan 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 ini 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 rincian

Bagian Rincian Build menampilkan build yang di-host di armada. Pilih nama build untuk melihat halaman detail build lengkap.

Konfigurasi runtime

Bagian konfigurasi Runtime menampilkan proses server yang akan diluncurkan pada setiap instance. Ini termasuk jalan untuk server game yang tidak dapat dieksekusi dan opsional parameter peluncuran.

Aktivasi sesi game

Bagian aktivasi sesi Game menampilkan jumlah proses server yang diluncurkan secara bersamaan dan berapa lama menunggu proses diaktifkan sebelum mengakhirinya.

Pengaturan port EC2

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 AmazonGameLift, lihatPantau Amazon GameLift dengan Amazon CloudWatch.

Kejadian

Tab Peristiwa menyediakan log dari semua peristiwa yang telah terjadi pada armada, termasuk kode peristiwa, pesan, dan stempel waktu. Lihat Deskripsi peristiwa di Referensi GameLift API Amazon.

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 lebih lanjut tentang perubahan batas dan kapasitas, lihatPenskalaan kapasitas GameLift hosting Amazon.

- AWSLokasi - Nama lokasi tempat instance armada digunakan.
- 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 instance aktif untuk mempertahankan lokasi. Ketika instance aktif dan instans yang diinginkan tidak sama, peristiwa penskalaan mulai memulai atau mematikan instance sesuai kebutuhan hingga instance aktif sama dengan instance yang diinginkan.
- Ukuran maks - Sebagian besar instance yang dapat digunakan di lokasi.
- Tersedia — Batas layanan pada instans dikurangi jumlah instance yang digunakan. Nilai ini memberitahu Anda jumlah maksimum instans yang dapat Anda tambahkan ke lokasi.

Kebijakan Auto-scaling

Bagian ini mencakup informasi tentang kebijakan penskalaan otomatis yang diterapkan pada armada. Anda dapat mengatur atau memperbarui kebijakan berbasis target. Kebijakan berbasis aturan armada, yang harus didefinisikan menggunakan SDK atau CLI AWS, ditampilkan di sini. Untuk informasi selengkapnya tentang penskalaan, lihat Kapasitas armada skala otomatis dengan Amazon GameLift.

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.
- Instans aktif - Jumlah instance dengan proses server yang berjalan di lokasi armada.
- Server aktif - Jumlah proses server game yang dapat menyelenggarakan 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 game untuk mengakses informasi sesi permainan tambahan, termasuk sesi pemain. Untuk informasi selengkapnya tentang sesi pemain, lihatMelihat data pada sesi game dan pemain.

Melihat data pada sesi game dan pemain

Anda dapat melihat informasi tentang sesi permainan dan pemain individu. Untuk informasi lebih lanjut tentang sesi game dan sesi pemain, lihat Bagaimana pemain terhubung ke game.

Untuk melihat sesi game dan data pemain

1. Di GameLiftkonsol Amazon, di panel navigasi, pilih Armada.
2. Pilih armada dari daftar Armada yang menyelenggarakan sesi permainan Anda.
3. Pilih tab Sesi permainan. Tab ini mencantumkan semua sesi game yang di-host di armada beserta informasi ringkasan.
4. Pilih sesi permainan untuk melihat informasi tambahan tentang sesi permainan dan daftar pemain yang terhubung ke game.

Detail

Gambaran Umum

Bagian ini menampilkan ringkasan informasi sesi permainan Anda.

- Status – Status sesi game.
  - Mengaktifkan - Instance memulai sesi permainan.
  - Aktif - Sesi permainan berjalan dan tersedia untuk menerima pemain, tergantung pada kebijakan pembuatan pemain sesi.
  - Dihentikan - sesi permainan telah berakhir.

- ARN - Nama Sumber Daya Amazon dari sesi game.
- Nama - Nama yang dihasilkan untuk sesi permainan.
- Lokasi - Lokasi di mana Amazon GameLift menjadi tuan rumah sesi game.
- Waktu pembuatan — Tanggal dan waktu Amazon GameLift membuat sesi streaming.
- Waktu akhir - Tanggal dan waktu sesi permainan berakhir.
- Nama DNS - Nama host 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 yang ditetapkan pada semua instans baru yang GameLift dimulai Amazon dalam armada.

Data permainan

Data yang diformat dengan baik untuk dikirim ke sesi permainan Anda saat mulai.

Properti game

Properti pasangan kunci dan nilai yang memengaruhi sesi game Anda.

Data perjodohan

JSON mak FlexMatch comblang. Untuk meninjau dan mengedit mak comblang pilih Lihat konfigurasi perjodohan. Untuk informasi lebih lanjut tentang FlexMatch perjodohan, lihat Membangun mak comblang.

Sesi pemain

Data sesi pemain berikut dikumpulkan untuk setiap sesi game:

- ID sesi pemain - Pengenal yang ditetapkan ke sesi pemain.
- ID Pemain – Pengidentifikasi unik untuk pemain. Pilih ID ini untuk mendapatkan informasi pemain tambahan.
- Status – Status dari sesi pemain. Berikut ini adalah status yang mungkin:

  - Reserved - Sesi pemain telah dipesan, tetapi para pemain tidak terhubung.
  - Aktif - Sesi pemain terhubung ke server game.
  - Selesai – Sesi pemain telah berakhir; pemain tidak lagi terhubung.
  - Waktunya Habis - Pemain gagal terhubung.
- Waktu pembuatan - Waktu pemain terhubung ke sesi permainan.
- Waktu akhir - Waktu pemain terputus dari sesi permainan.
- Data pemain - Informasi tentang pemain yang disediakan selama pembuatan sesi pemain.

Informasi pemain

Lihat informasi tambahan untuk pemain yang terpilih, termasuk daftar semua game yang terhubung dengan pemain di semua armada di wilayah saat ini. Informasi ini mencakup status, waktu mulai, waktu akhir, dan total waktu terhubung untuk setiap sesi pemain. Anda dapat memilih untuk melihat data untuk sesi dan armada game yang relevan.

Lihat alias Anda

Halaman Alias menampilkan informasi tentang alias armada yang Anda buat di Wilayah Anda saat ini. Untuk melihat halaman alias, pilih Alias di panel navigasi.

Anda dapat melakukan hal berikut pada halaman alias:

- Buat alias baru. Pilih Buat alias.
- Filter dan urutkan tabel alias. Gunakan kontrol di bagian atas tabel.
- Melihat detail alias. Pilih nama alias untuk membuka halaman detail alias.
- Menghapus alias. Pilih alias dan kemudian pilih Hapus.

Detail alias

Halaman rincian alias menampilkan informasi tentang alias.

Dari halaman ini Anda dapat:

- Mengedit alias. Pilih Edit.

- Lihat armada yang Anda kaitkan dengan alias.
- Menghapus alias. Pilih Delete (Hapus).

Informasi detail alias meliputi:

- ID - Nomor unik yang digunakan untuk mengidentifikasi alias.
- Deskripsi – Deskripsi fungsi.
- ARN — Nama Sumber Daya Amazon alias.
- Penciptaan - Tanggal dan waktu alias dibuat.
- Terakhir diperbarui - Tanggal dan waktu alias terakhir diperbarui.
- Jenis perutean - Jenis perutean untuk alias, yang bisa menjadi salah satunya:
  - Sederhana - Rute lalu lintas pemain ke ID armada tertentu. Anda dapat memperbarui ID armada untuk alias kapan saja.
  - Terminal - Melewati pesan kembali ke klien. Misalnya, Anda dapat mengarahkan pemain yang menggunakan out-of-date klien ke lokasi di mana mereka bisa mendapatkan peningkatan.
- Tags - Pasangan kunci dan nilai yang digunakan untuk mengidentifikasi alias.

Melihat antrean

Anda dapat melihat informasi tentang semua antrean penempatan sesi game yang ada. Halaman antrian menampilkan antrian yang dibuat di Wilayah Anda saat ini. 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. Menyiapkan GameLift antrean Amazon untuk penempatan sesi game

Halaman antrian menampilkan informasi ringkasan berikut untuk setiap antrian:

- 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 menempatkan sesi permainan baru pada armada apa pun dalam antrian.

Lihat detail antrean

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.
- Menghapus antrian. Setelah Anda menghapus antrian, semua permintaan untuk sesi game baru yang merujuk bahwa nama antrian akan gagal. Pilih Delete (Hapus).

> **Note**
>
> Untuk mengembalikan antrian yang dihapus, buat antrian baru dengan nama antrian yang dihapus.

Detail

Gambaran Umum

Bagian Ikhtisar menampilkan Amazon Resource Name (ARN) antrean dan Timeout. Anda dapat menggunakan ARN saat mereferensikan antrian di tindakan atau area lain di Amazon. GameLift Batas waktu adalah durasi waktu maksimum, dalam hitungan detik, bahwa permintaan penempatan sesi game tetap dalam antrian sebelum waktu habis.

Notifikasi peristiwa

Bagian pemberitahuan peristiwa mencantumkan topik SNS Amazon GameLift menerbitkan pemberitahuan peristiwa 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 pemberian tag, lihat Menandai sumber daya AWS.

Metrik

Tab Metrik menampilkan representasi grafis metrik antrean dari waktu ke waktu.

Metrik antrean mencakup berbagai informasi yang menjelaskan aktivitas penempatan di seluruh antrean, termasuk penempatan berhasil yang diatur berdasarkan Wilayah. Anda dapat menggunakan data Wilayah untuk memahami di mana Anda meng-hosting game Anda. Metrik penempatan regional dapat membantu mendeteksi masalah dengan desain antrian keseluruhan.

Metrik antrean juga tersedia di Amazon. CloudWatch Untuk deskripsi metrik yang tersedia, lihat. GameLiftMetrik Amazon untuk antrean

Tujuan

Tab Tujuan menampilkan semua armada atau alias yang terdaftar untuk antrean.

Saat Amazon GameLift menelusuri tujuan sumber daya yang tersedia untuk menjadi tuan rumah sesi game baru, Amazon akan mencari urutan default yang tercantum di sini. Selama ada kapasitas di tujuan pertama yang terdaftar, Amazon GameLift 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 untuk mencari tujuan yang tersedia dengan latensi pemain rata-rata terendah. Untuk informasi selengkapnya tentang mendesain antrean Anda, lihat. Desain antrean sesi game

Penempatan sesi

Kebijakan latensi pemain

Bagian Kebijakan latensi pemain menunjukkan semua kebijakan yang digunakan antrean. Tabel mencantumkan kebijakan dalam urutan mereka ditegakkan.

Lokasi

Bagian Lokasi menunjukkan lokasi di mana antrian ini dapat menempatkan sesi permainan.

Priority

Bagian Prioritas menunjukkan urutan antrian mengevaluasi detail sesi game.

Urutan lokasi

Bagian Urutan lokasi menunjukkan urutan default yang digunakan antrian saat menempatkan sesi game. Antrian menggunakan urutan ini jika Anda belum menentukan jenis prioritas lainnya.

Memantau Amazon GameLift

Jika Anda menggunakan Amazon GameLift FleetiQ sebagai fitur mandiri dengan Amazon EC2, lihat Keamanan di Amazon EC2 di Panduan Pengguna Amazon EC2 untuk Instans Linux.

Pemantauan adalah bagian penting dalam menjaga keandalan, ketersediaan, dan kinerja Amazon GameLift dan AWS solusi Anda yang lain. Ada tiga kegunaan utama untuk metrik dengan Amazon GameLift: 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, melaporkan ketika ada sesuatu yang salah, dan mengambil tindakan otomatis bila perlu:

- GameLift Konsol Amazon
- Amazon CloudWatch —— Anda dapat memantau GameLift metrik Amazon 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 untuk Amazon GameLift 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

Pantau Amazon GameLift dengan Amazon CloudWatch

Anda dapat memantau Amazon GameLift menggunakan AmazonCloudWatch, AWS layanan yang mengumpulkan data mentah dan memprosesnya menjadi mudah dibaca, mendekati metrik waktu nyata. Statistik ini disimpan selama 15 bulan untuk memberikan perspektif historis tentang bagaimana server game Anda hosting dengan Amazon GameLift berkinerja. Anda dapat mengatur alarm yang mengawasi ambang batas tertentu dan mengirim pemberitahuan atau mengambil tindakan saat ambang batas tersebut terpenuhi. Untuk informasi lebih lanjut, lihat Panduan Pengguna Amazon CloudWatch.

Tabel berikut mencantumkan metrik dan dimensi untuk AmazonGameLift. Semua metrik yang tersedia juga CloudWatch tersedia di GameLift konsol Amazon, yang menyediakan data sebagai sekumpulan grafik yang dapat disesuaikan. Untuk mengakses CloudWatch metrik game Anda, gunakan APIAWS CLI, atau CloudWatch API. AWS Management Console

Jika metrik tidak memiliki lokasi, metrik menggunakan lokasi rumah.

Dimensi untuk GameLift metrik Amazon

Amazon GameLift mendukung metrik pemfilteran berdasarkan dimensi berikut.

Dimensi	Deskripsi
`Location`	Filter metrik untuk lokasi penyebaran armada. Jika metrik tidak memiliki lokasi, metrik 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.

Dimensi	Deskripsi
QueueName	Filter metrik untuk antrean tunggal. Dimensi ini hanya digunakan dengan metrik untuk antrean sesi game.
ConfigurationName	Filter metrik untuk konfigurasi matchmaking tunggal. Dimensi ini digunakan dengan metrik untuk konfigura si matchmaking.
ConfigurationName-RuleName	Filter metrik untuk perpotongan konfigurasi matchmaking dan aturan matchmaking. Dimensi ini digunakan dengan metrik untuk aturan matchmaking saja.
InstanceType	Filter metrik untuk menunjukan tipe instans EC2, 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.

GameLiftMetrik Amazon untuk armada

Namespace AWS/GameLift mencakup metrik berikut yang terkait dengan aktivitas di seluruh armada atau grup armada. Armada digunakan dengan GameLift solusi Amazon yang dikelola. GameLiftLayanan Amazon 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

Metrik	Deskripsi
	mengukur total kapasitas instans saat ini. Metrik ini dapat digunakan dengan penskalaan otomatis. Unit: Count (Jumlah) CloudWatchStatistik yang relevan: Rata-rata, Minimum, Maksimum Dimensi: Lokasi
`DesiredInstances`	Targetkan jumlah instans aktif yang GameLift dipelihara Amazon dalam armada. Dengan penskalaan otomatis, nilai ini ditentukan 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: Count (Jumlah) CloudWatchStatistik 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: Count (Jumlah) CloudWatchStatistik yang relevan: Rata-rata, Minimum, Maksimum Dimensi: Lokasi

Metrik	Deskripsi
`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: Count (Jumlah) CloudWatchStatistik yang relevan: Rata-rata, Minimum, Maksimum Dimensi: Lokasi
`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: Count (Jumlah) CloudWatchStatistik yang relevan: Rata-rata, Minimum, Maksimum Dimensi: Lokasi
`PercentIdleInstances`	Persentase dari semua instans aktif yang siaga (dihitung sebagai `IdleInstances / ActiveInstances`). Metrik ini dapat digunakan untuk penskalaan otomatis. Unit: Persen CloudWatchStatistik yang relevan: Rata-rata, Minimum, Maksimum Dimensi: Lokasi

Metrik	Deskripsi
`RecycledInstances`	Jumlah instans spot yang telah didaur ulang dan diganti. Amazon GameLift mendaur ulang instans spot yang saat ini tidak menghosting sesi game dan memiliki kemungkinan gangguan yang tinggi. Unit: Count (Jumlah) CloudWatchStatistik yang relevan: Jumlah, Rata-rata , Minimum, Maksimum Dimensi: Lokasi
`InstanceInterruptions`	Jumlah dari instans spot yang telah terganggu. Unit: Count (Jumlah) CloudWatchStatistik yang relevan: Jumlah, Rata-rata , Minimum, Maksimum Dimensi: Lokasi
`CPUUtilization`	Metrik EC2. Untuk Amazon, metrik GameLift ini merepresentasikan kinerja perangkat keras di semua instans aktif di lokasi armada. Persentas e waktu CPU fisik yang digunakan Amazon EC2 untuk menjalankan instans, yang mencakup waktu yang dihabiskan untuk menjalankan kode pengguna dan kode Amazon EC2. Alat dalam sistem operasi Anda dapat menunjukkan persentase yang berbeda dibandingkan CloudWatch dengan faktor-faktor seperti simulasi perangkat lama, konfigurasi perangkat non-warisan, beban kerja interupsi berat, migrasi langsung, dan pembaruan langsung. Unit: Persen

Metrik	Deskripsi
NetworkIn	Metrik EC2. Untuk Amazon, metrik GameLift ini merepresentasikan 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: Bita
NetworkOut	Metrik EC2. Untuk Amazon, metrik GameLift ini merepresentasikan kinerja perangkat keras di semua instans aktif di lokasi armada. Jumlah byte yang dikirimkan keluar pada semua antarmuka jaringan oleh instans. Metrik ini mengidentifikasi volume lalu lintas jaringan keluar ke aplikasi pada satu instance. Unit: Bita
DiskReadBytes	Metrik EC2. Untuk Amazon, metrik GameLift ini merepresentasikan kinerja perangkat keras di semua instans aktif di lokasi armada. Byte 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: Bita

Metrik	Deskripsi
DiskWriteBytes	Metrik EC2. Untuk Amazon, metrik GameLift ini merepresentasikan kinerja perangkat keras di semua instans aktif di lokasi armada. Byte 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: Bita
DiskReadOps	Metrik EC2. Untuk Amazon, metrik GameLift ini merepresentasikan 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 tertentu. Untuk menghitung operasi I/O rata-rata per detik atau I/O operations per second (IOPS) untuk periode tersebut, bagi total operasi dalam periode dengan jumlah detik dalam periode tersebut. Unit: Count (Jumlah)
DiskWriteOps	Metrik EC2. Untuk Amazon, metrik GameLift ini merepresentasikan kinerja perangkat keras di semua instans aktif di lokasi armada. Operasi tulis yang telah diselesaikan ke semua volume penyimpanan instans yang tersedia untuk instans tersebut dalam periode waktu tertentu. Untuk menghitung operasi I/O rata-rata per detik atau I/O operations per second (IOPS) untuk periode tersebut, bagi total operasi dalam periode dengan jumlah detik dalam periode tersebut. Unit: Count (Jumlah)

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: Count (Jumlah) CloudWatchStatistik 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: Count (Jumlah) CloudWatchStatistik yang relevan: Rata-rata, Minimum, Maksimum Dimensi: Lokasi
`PercentHealthyServerProcesses`	Persentase semua proses server aktif yang melaporkan kondisi (dihitung sebagai `HealthyServerProcesses / ActiveServerProcesses`). Unit: Persen CloudWatchStatistik 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 AmazonGameLift. Ini terjadi ketika proses server berhenti merespons, secara konsisten melaporkan pemeriksaan kesehatan yang gagal, atau tidak berakhir dengan bersih (dengan memanggil ProcessEnding ()). Unit: Count (Jumlah) CloudWatchStatistik 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: Count (Jumlah) CloudWatchStatistik 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: Count (Jumlah) CloudWatchStatistik 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 sesu game tidak bertransisi dari MENGAKTIFKAN ke status AKTIF. Metrik ini dapat digunakan dengan penskalaan otomatis. Unit: Count (Jumlah) CloudWatchStatistik 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: Count (Jumlah) CloudWatchStatistik yang relevan: Rata-rata, Minimum, Maksimum Dimensi: Lokasi
`AvailableGameSessions`	Proses server yanf 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 `ConcurrentActivatableGameSessions` . Metrik tersebut lebih akurat mewakili jumlah sesi game baru yang dapat dimulai tanpa penundaan apa pun. Unit: Count (Jumlah) CloudWatchStatistik yang relevan: Rata-rata, Minimum, Maksimum Dimensi: Lokasi

Metrik	Deskripsi
`ConcurrentActivatableGameSessions`	Proses server yanf aktif dan sehat yang saat ini tidak digunakan untuk meng-host sesi game dan dapat segera memulai sesi game baru. Metrik ini berbeda dari `AvailableGameSessions` dengan cara berikut: tidak menghitung proses server yang saat ini tidak dapat mengaktifkan sesi game baru karena batas pada aktivasi sesi game. (Lihat pengaturan RuntimeConfigurationopsional armada`MaxConcurrentGameSessionActivations`). Untuk armada yang tidak membatasi aktivasi sesi game, metrik ini identik dengan `AvailableGameSessions` . Unit: Count (Jumlah) CloudWatchStatistik yang relevan: Rata-rata, Minimum, Maksimum Dimensi: Lokasi
`PercentAvailableGameSessions`	Persentase slot sesi game pada semua proses server aktif (sehat atau tidak sehat) yang saat ini tidak sedang digunakan (dihitung sebagai `AvailableGameSessions / [ActiveGameSessions + AvailableGameSessions + unhealthy server processes]`). Metrik ini dapat digunakan dengan penskalaan otomatis. Unit: Persen CloudWatchStatistik yang relevan: Rata-rata Dimensi: Lokasi

Metrik	Deskripsi
GameSessionInterruptions	Jumlah sesi game pada instans spot yang telah terganggu. Unit: Count (Jumlah) CloudWatchStatistik 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: Count (Jumlah) CloudWatchStatistik 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: Count (Jumlah) CloudWatchStatistik yang relevan: Jumlah, Rata-rata , Minimum, Maksimum

GameLiftMetrik Amazon untuk antrean

Namespace `Amazon GameLift` mencakup metrik berikut yang terkait dengan aktivitas di antrean penempatan sesi game. Antrian digunakan dengan solusi Amazon GameLift yang dikelola. GameLiftLayanan Amazon mengirimkan metrik ke CloudWatch setiap menit.

Metrik	Deskripsi
`AverageWaitTime`	Jumlah rata-rata waktu bahwa permintaan penempatan sesi game dalam antrean dengan status TERTUNDA telah menunggu untuk dipenuhi. Unit: detik CloudWatchStatistik 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 tercantum dalam antrean atau—ketika permintaa n penempatan mencakup data latensi pemain—in i adalah armada pertama yang dipilih oleh prioritas FleetIQ. Jika tidak ada armada spot yang layak, setiap armada di wilayah tersebut dapat dipilih. Unit: Count (Jumlah) CloudWatchStatistik 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

Metrik	Deskripsi
	tersedia. Armada pilihan pertama adalah armada pertama yang tercantum dalam antrian atau—ketika permintaan penempatan menyertakan data latensi pemain —ini adalah armada pertama yang dipilih oleh prioritas FleetiQ yang Anda tentukan. Unit: Count (Jumlah) CloudWatchStatistik yang relevan: Rata-rata, Minimum, Maksimum, Jumlah
`LowestLatencyPlacement`	Sesi game yang berhasil ditempatkan di wilayah yang menawarkan latensi serendah mungkin antrean untuk para pemain. Metrik ini dipancarkan hanya bila data latensi pemain disertakan dalam permintaan penempatan. Unit: Count (Jumlah) CloudWatchStatistik yang relevan: Rata-rata, Minimum, Maksimum, Jumlah
`LowestPricePlacement`	Sesi game berhasil ditempatkan dalam armada dengan harga antrean serendah mungkin untuk wilayah yang dipilih. Armada ini dapat berupa armada spot atau instans sesuai permintaan jika antrean tidak memiliki instans spot. Metrik ini dipancarkan hanya bila data latensi pemain disertaka n dalam permintaan penempatan. Unit: Count (Jumlah) CloudWatchStatistik 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: Count (Jumlah) CloudWatchStatistik yang relevan: Jumlah
`PlacementsCanceled`	Permintaan penempatan sesi game yang dibatalkan sebelum waktu habis sejak laporan terakhir. Unit: Count (Jumlah) CloudWatchStatistik yang relevan: Rata-rata, Minimum, Maksimum, Jumlah
`PlacementsFailed`	Permintaan penempatan sesi game yang gagal karena alasan apapun sejak laporan terakhir. Unit: Count (Jumlah) CloudWatchStatistik yang relevan: Rata-rata, Minimum, Maksimum, Jumlah
`PlacementsStarted`	Permintaan penempatan sesi game baru yang ditambahkan ke antrean sejak laporan terakhir. Unit: Count (Jumlah) CloudWatchStatistik yang relevan: Rata-rata, Minimum, Maksimum, Jumlah

Metrik	Deskripsi
PlacementsSucceeded	Permintaan penempatan sesi game yang menghasilkan sesi game baru sejak laporan terakhir. Unit: Count (Jumlah) CloudWatchStatistik yang relevan: Rata-rata, Minimum, Maksimum, Jumlah
PlacementsTimedOut	Permintaan penempatan sesi game yang mencapai batas waktu antrean tanpa terpenuhi sejak laporan terakhir. Unit: Count (Jumlah) CloudWatchStatistik yang relevan: Rata-rata, Minimum, Maksimum, Jumlah
QueueDepth	Jumlah permintaan penempatan sesi game dalam antrean dengan status TERTUNDA. Unit: Count (Jumlah) CloudWatchStatistik yang relevan: Rata-rata, Minimum, Maksimum, Jumlah Dimensi: Lokasi

GameLiftMetrik Amazon untuk perjodohan

`Amazon GameLift`Namespace mencakup metrik FlexMatch aktivitas untuk konfigurasi perjodohan dan aturan perjodohan. FlexMatchperjodohan digunakan dengan solusi Amazon GameLift yang dikelola. GameLiftLayanan Amazon mengirimkan metrik ke CloudWatch setiap menit.

Untuk informasi selengkapnya tentang urutan aktivitas perjodohan, lihat Cara kerja Amazon GameLift FlexMatch.

Konfigurasi matchmaking

Metrik	Deskripsi
`CurrentTickets`	Permintaan matchmaking saat ini sedang diproses atau menunggu untuk diproses. Unit: Count (Jumlah) CloudWatchStatistik yang relevan: Rata-rata, Minimum, Maksimum, Jumlah
`MatchAcceptancesTimedOut`	Untuk konfigurasi matchmaking yang memerluka n penerimaan, pertandingan potensial yang habis waktunya selama penerimaan sejak laporan terakhir. Unit: Count (Jumlah) CloudWatchStatistik yang relevan: Jumlah
`MatchesAccepted`	Untuk konfigurasi matchmaking yang memerlukan penerimaan, pertandingan potensial yang diterima sejak laporan terakhir. Unit: Count (Jumlah) CloudWatchStatistik yang relevan: Jumlah
`MatchesCreated`	Potensi pertandingan yang dibuat sejak laporan terakhir. Unit: Count (Jumlah) CloudWatchStatistik yang relevan: Jumlah
`MatchesPlaced`	Pertandingan yang berhasil ditempatkan ke dalam sesi game sejak laporan terakhir. Unit: Count (Jumlah) CloudWatchStatistik yang relevan: Jumlah

Metrik	Deskripsi
MatchesRejected	Untuk konfigurasi matchmaking yang memerluka n penerimaan, pertandingan potensial yang ditolak oleh setidaknya satu pemain sejak laporan terakhir. Unit: Count (Jumlah) CloudWatchStatistik yang relevan: Jumlah
PlayersStarted	Pemain dalam tiket matchmaking yang ditambahkan sejak laporan terakhir. Unit: Count (Jumlah) CloudWatchStatistik yang relevan: Jumlah
TicketsFailed	Permintaan matchmaking yang mengakibatkan kegagalan sejak laporan terakhir. Unit: Count (Jumlah) CloudWatchStatistik yang relevan: Jumlah
TicketsStarted	Permintaan matchmaking baru yang dibuat sejak laporan terakhir. Unit: Count (Jumlah) CloudWatchStatistik yang relevan: Jumlah
TicketsTimedOut	Permintaan matchmaking yang mencapai batas waktu habis sejak laporan terakhir. Unit: Count (Jumlah) CloudWatchStatistik 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 CloudWatchStatistik 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 CloudWatchStatistik 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 CloudWatchStatistik 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: Count (Jumlah) CloudWatchStatistik yang relevan: Jumlah
`RuleEvaluationsFailed`	Evaluasi aturan selama matchmaking yang gagal sejak laporan terakhir. Metrik ini terbatas pada 50 aturan teratas. Unit: Count (Jumlah) CloudWatchStatistik yang relevan: Jumlah

GameLiftMetrik Amazon untuk FleetiQ

Namespace `Amazon GameLift` termasuk metrik untuk grup server game FleetIQ dan aktivitas server game sebagai bagian dari solusi mandiri FleetIQ untuk hosting game. GameLiftLayanan Amazon mengirimkan metrik ke CloudWatch setiap menit. Lihat juga Memantau grup dan instans Penskalaan Otomatis Anda menggunakan amazon CloudWatch di Panduan Pengguna Penskalaan Otomatis Amazon EC2.

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: Count (Jumlah) CloudWatchStatistik 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: Count (Jumlah) CloudWatchStatistik 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: Count (Jumlah) CloudWatchStatistik yang relevan: Jumlah Dimensi: GameServerGroup
`DrainingUtilizedGameServers`	Server game pada instans yang dijadwalkan untuk dihentikan yang saat ini men-support gameplay. Unit: Count (Jumlah) CloudWatchStatistik 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 CloudWatchStatistik yang relevan: Rata-rata, Minimum, Maksimum Dimensi: GameServerGroup
`GameServerInterruptions`	Server game di Instans spot yang terganggu karena ketersediaan Spot terbatas. Unit: Count (Jumlah) CloudWatchStatistik yang relevan: Jumlah Dimensi:GameServerGroup, InstanceType
`InstanceInterruptions`	Instans Spot yang terganggu karena ketersediaan yang terbatas. Unit: Count (Jumlah) CloudWatchStatistik yang relevan: Jumlah Dimensi:GameServerGroup, InstanceType

Mencatat panggilan GameLift API Amazon dengan AWS CloudTrail

Amazon GameLift terintegrasi denganAWS CloudTrail, layanan yang menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau AWS layanan di AmazonGameLift. CloudTrailmenangkap

semua panggilan API untuk Amazon GameLift sebagai peristiwa. Panggilan yang diambil termasuk panggilan dari GameLift konsol Amazon dan panggilan kode ke operasi Amazon GameLift API. Jika Anda membuat jejak, Anda dapat mengaktifkan pengiriman CloudTrail acara berkelanjutan ke bucket Amazon S3, termasuk acara untuk AmazonGameLift. Jika Anda tidak membuat konfigurasi jejak, Anda masih dapat melihat kejadian terbaru dalam konsol CloudTrail di Riwayat peristiwa. Menggunakan informasi yang dikumpulkan olehCloudTrail, Anda dapat menentukan permintaan yang dibuat untuk AmazonGameLift, alamat IP dari mana permintaan dibuat, yang membuat permintaan, ketika dibuat, dan rincian tambahan.

Untuk mempelajari lebih lanjutCloudTrail, lihat Panduan AWS CloudTrail Pengguna.

GameLiftInformasi Amazon di CloudTrail

CloudTraildiaktifkan pada Akun AWS saat Anda membuat akun. Ketika aktivitas terjadi di AmazonGameLift, aktivitas tersebut direkam dalam CloudTrail peristiwa bersama dengan peristiwa AWS layanan lainnya dalam riwayat peristiwa. Anda dapat melihat, mencari, dan mengunduh peristiwa terbaru di Akun AWS Anda. Untuk informasi selengkapnya, lihat Melihat peristiwa dengan Riwayat CloudTrail peristiwa.

Untuk catatan peristiwa yang sedang berlangsung di acara AndaAkun AWS, termasuk acara untuk AmazonGameLift, buat jejak. Jejak memungkinkan CloudTrail untuk mengirim berkas 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 partisi AWS dan mengirimkan berkas log ke bucket Amazon S3 yang Anda tentukan. Selain itu, Anda dapat membuat konfigurasi layanan AWS lainnya untuk menganalisis lebih lanjut dan bertindak berdasarkan data peristiwa yang dikumpulkan di log CloudTrail. Untuk informasi selengkapnya, lihat yang berikut:

- Ikhtisar untuk membuat jejak
- CloudTraillayanan dan integrasi yang didukung
- Mengonfigurasi notifikasi Amazon SNS untuk CloudTrail
- Menerima file log CloudTrail dari beberapa wilayah dan Menerima file log CloudTrail dari beberapa akun

Semua GameLift tindakan Amazon dicatat oleh CloudTrail dan didokumentasikan dalam Referensi GameLift API Amazon. Misalnya, panggilan ke`CreateGameSession`, `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 hal berikut:

- Bahwa permintaan dibuat dengan kredensial pengguna root atau pengguna AWS Identity and Access Management (IAM).
- Bahwa permintaan tersebut dibuat dengan kredensial keamanan sementara untuk peran atau pengguna gabungan.
- Bahwa permintaan dibuat oleh layanan AWS lain.

Untuk informasi selengkapnya, lihat Elemen userIdentity CloudTrail.

Memahami entri file GameLift log Amazon

Jejak adalah konfigurasi yang memungkinkan pengiriman peristiwa sebagai berkas log ke bucket Amazon S3 yang telah Anda tentukan. Berkas log CloudTrail berisi satu atau beberapa entri log. Peristuwa mewakili satu permintaan dari sumber apa pun dan mencakup informasi tentang tindakan yang diminta, tanggal dan waktu tindakan, parameter permintaan, dan sebagainya. Berkas log CloudTrail bukan jejak tumpukan terurut dari panggilan API publik, sehingga berkas 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

Anda dapat menangkap pesan server khusus dari GameLift server Amazon Anda dalam file log. Cara Anda mengkonfigurasi logging tergantung pada apakah Anda menggunakan server kustom atau Server Realtime (lihat subbagian yang sesuai di bagian ini).

Topik

- Pencatatan pesan server (server khusus)
- Mencatat pesan server (Server Realtime)

Pencatatan pesan server (server khusus)

Anda dapat menangkap pesan server khusus dari server GameLift kustom Amazon Anda dalam file log. Untuk mempelajari tentang logging untuk Server Realtime, lihatMencatat pesan server (Server Realtime).

> **Important**
>
> Ada batasan ukuran file log per sesi game (lihat GameLift titik akhir Amazon dan kuota di Referensi Umum AWS). Saat sesi permainan berakhir, Amazon GameLift mengunggah log server ke Amazon Simple Storage Service (Amazon S3). Amazon tidak GameLift 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 GameLift kustom Amazon, Anda menulis kode Anda sendiri untuk melakukan logging, yang Anda konfigurasikan sebagai bagian dari konfigurasi proses server Anda. Amazon GameLift menggunakan konfigurasi logging Anda untuk mengidentifikasi file yang harus diunggah ke Amazon S3 di akhir setiap sesi game.

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 LogParametersProcessParametersobjek 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 ProcessParametersobjek saat Anda memanggil ProcessReady().

```
Aws::GameLift::GenericOutcome outcome =
   Aws::GameLift::Server::ProcessReady(processReadyParameter);
```

Untuk contoh yang lebih lengkap, lihatProcessReady().

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 LogParametersProcessParametersobjek Anda.

```
var processReadyParameter = new ProcessParameters(
    this.OnGameSession,
    this.OnProcessTerminate,
    this.OnHealthCheck,
    this.OnGameSessionUpdate,
    port,
    new LogParameters(logPaths));
```

3. Berikan ProcessParametersobjek saat Anda memanggil ProcessReady().

```
var processReadyOutcome =
   GameLiftServerAPI.ProcessReady(processReadyParameter);
```

Untuk contoh yang lebih lengkap, lihatProcessReady().

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 secara otomatis menyimpan log di ember Amazon S3 dan menyimpannya selama 14 hari. Untuk mendapatkan lokasi log untuk sesi game, Anda dapat menggunakan operasi GetGameSessionLogUrlAPI. Untuk mengunduh log, gunakan URL yang dikembalikan operasi.

Mencatat pesan server (Server Realtime)

Anda dapat menangkap pesan server kustom dari Server Realtime Anda dalam file log. Untuk mempelajari tentang pencatatan untuk server kustom, lihatPencatatan pesan server (server khusus).

Ada berbagai jenis pesan yang dapat Anda ouptput ke file log Anda (lihatMencatat pesan di skrip server Anda). Selain pesan kustom Anda, Server Realtime Anda mengeluarkan pesan sistem 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 (lihatMenyesuaikan level logging).

> **Important**
>
> Ada batasan ukuran file log per sesi game (lihat GameLift titik akhir Amazon dan kuota di Referensi Umum AWS). Saat sesi permainan berakhir, Amazon GameLift mengunggah

log server ke Amazon Simple Storage Service (Amazon S3). Amazon tidak GameLift 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 Server Realtime Anda. Gunakan langkah-langkah berikut untuk mengirim pesan server ke file log:

1. Buat varaible 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 kesalahan

```
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, lihatContoh skrip Server Realtime.

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 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 Server Realtime, Anda dapat menyesuaikan level logging dengan menyediakan parameter dalam konfigurasi runtime armada Anda di formulir`loggingLevel:`*`LOGGING_LEVEL`*, yang *`LOGGING_LEVEL`* merupakan 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 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.

Menyetel level logging saat membuat armada Server Realtime (Konsol)

Ikuti petunjuk di Buat armada GameLift terkelola Amazon 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`

Menyetel level logging saat membuat armada Server Realtime () AWS CLI

Ikuti petunjuk di Buat armada GameLift terkelola Amazon 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 armada Server Realtime yang sedang berjalan (Konsol)

Ikuti petunjuk di Memperbarui konfigurasi armada untuk memperbarui armada Anda menggunakan GameLift Konsol Amazon, 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 armada Server Realtime yang sedang berjalan () AWS CLI

Ikuti petunjuk di Memperbarui konfigurasi armada untuk memperbarui armada Anda menggunakanAWS 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

Jika Anda menggunakan Amazon GameLift FleetiQ sebagai fitur mandiri dengan Amazon EC2, lihat Keamanan di Amazon EC2 di Panduan Pengguna Amazon EC2 untuk Instans Linux.

Keamanan cloud di AWS merupakan prioritas tertinggi. Sebagai pelanggan AWS, Anda mendapatkan manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan organisasi yang paling sensitif terhadap keamanan.

Keamanan adalah tanggung jawab bersama antara AWS dan Anda. Model tanggung jawab bersama menggambarkan ini sebagai keamanan dari cloud dan keamanan di dalam cloud:

- Keamanan cloud – AWS bertanggung jawab untuk melindungi infrastruktur yang menjalankan layanan-layanan AWS di dalam AWS Cloud. AWS juga memberikan Anda layanan yang dapat digunakan dengan aman. Auditor pihak ketiga menguji dan memverifikasi secara rutin efektivitas keamanan kami sebagai bagian dari program kepatuhan AWS. Untuk mempelajari tentang program kepatuhan yang berlaku untuk AmazonGameLift, lihat AWSlayanan dalam lingkup oleh AWS layanan program kepatuhan .
- Keamanan di cloud – Tanggung jawab Anda ditentukan menurut layanan AWS yang Anda gunakan. Anda juga bertanggung jawab atas faktor-faktor lain termasuk sensitivitas data Anda, persyaratan perusahaan Anda, dan l AWS dan peraturan yang berlaku.

Dokumentasi ini membantu Anda memahami cara menerapkan model tanggung jawab bersama saat menggunakan AmazonGameLift. Topik berikut menunjukkan cara mengonfigurasi Amazon GameLift untuk memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga mempelajari cara menggunakan AWS layanan lain yang membantu Anda memantau dan mengamankan GameLift sumber daya Amazon Anda.

Topik

- Analisis konfigurasi dan kerentanan di Amazon GameLift
- Praktik terbaik keamanan untuk Amazon GameLift

Perlindungan data di Amazon GameLift

Jika Anda menggunakan Amazon GameLift FleetiQ sebagai fitur mandiri dengan Amazon EC2, lihat Keamanan di Amazon EC2 di Panduan Pengguna Amazon EC2 untuk Instans Linux.

Model tanggung jawab AWS bersama model berlaku untuk perlindungan data di Amazon GameLift. Sebagaimana diuraikan dalam model ini, AWS bertanggung jawab untuk melindungi infrastruktur global yang menjalankan semua AWS Cloud. Anda bertanggung jawab untuk memelihara kendali atas isi yang dihost 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, sebaiknya lindungi kredensial Akun AWS dan siapkan untuk masing-masing pengguna 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 sumber daya AWS. Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- Siapkan API dan pengelogan aktivitas pengguna dengan AWS CloudTrail.
- Gunakan solusi enkripsi AWS, bersama semua kontrol keamanan bawaan dalam Layanan AWS.
- Gunakan layanan keamanan terkelola lanjut seperti Amazon Macie, yang membantu menemukan dan mengamankan data sensitif yang disimpan di Amazon S3.
- Jika Anda memerlukan modul kriptografi tervalidasi FIPS 140-2 ketika mengakses AWS melalui antarmuka baris perintah atau API, gunakan titik akhir FIPS. Lihat informasi yang lebih lengkap tentang titik akhir FIPS yang tersedia di Standar Pemrosesan Informasi Federal (FIPS) 140-2.

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 atau lainnya Layanan AWS menggunakan konsol, APIAWS CLI, atau AWS SDK. 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.

Data GameLift khusus Amazon ditangani sebagai berikut:

- Build server game dan skrip yang Anda unggah ke Amazon GameLift 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 GameLift konsol Amazon 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 GameLift konsol Amazon atau melalui panggilan ke API layanan.
- Data metrik dan peristiwa disimpan di Amazon GameLift dan dapat diakses melalui GameLift konsol Amazon atau melalui 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 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 ID pemain kustom dalam permintaan Anda, diharapkan bahwa nilai-nilai ini adalah UUID anonim dan tidak berisi informasi yang mengidentifikasi pemain.

Untuk informasi selengkapnya tentang perlindungan data, lihat postingan blog model tanggung tawab bersama AWS dan Peraturan Perlindungan Data Umum (GDPR) di Blog Keamanan AWS.

Enkripsi saat tidak digunakan

Enkripsi GameLift AT-rest dari data khusus Amazon ditangani sebagai berikut:

- Pembuatan dan skrip server game disimpan di bucket Amazon S3 dengan enkripsi sisi server.
- Data yang disediakan pelanggan disimpan di Amazon GameLift dalam format terenkripsi.

Enkripsi dalam bergerak

Koneksi ke GameLift API Amazon dibuat melalui koneksi aman (SSL) dan diautentikasi menggunakan AWSSignature 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 di GameLift sumber daya Amazon, komunikasi tidak melibatkan GameLift layanan Amazon. 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.
- Untuk Realtime Server dengan pembuatan sertifikat TLS diaktifkan, lalu lintas antara klien game dan server Realtime menggunakan Realtime Client SDK dienkripsi dalam penerbangan. Lalu lintas TCP dienkripsi menggunakan TLS 1.2, dan lalu lintas UDP dienkripsi menggunakan DTLS 1.2.

Privasi lalu lintas jaringan internet

Anda dapat mengakses GameLift instans Amazon 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 FleetiQ, 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 koneksi ke GameLift instans Amazon terkelola, lihatTerhubung dari jarak jauh ke instance GameLift armada Amazon.

Manajemen identitas dan akses untuk Amazon GameLift

AWS Identity and Access Management (IAM) adalah Layanan AWS yang membantu administrator mengontrol akses ke sumber daya AWS secara aman. Administrator IAM mengontrol siapa yang dapat diautentikasi (masuk) dan diotorisasi (memiliki izin) untuk menggunakan sumber daya Amazon.

GameLift IAM adalah layanan Layanan AWS yang dapat Anda gunakan tanpa dikenakan biaya tambahan.

Topik

Audiens

Cara Anda menggunakan AWS Identity and Access Management (IAM) berbeda, tergantung pada pekerjaan yang Anda lakukan di Amazon GameLift.

Pengguna layanan — Jika Anda menggunakan GameLift layanan Amazon untuk melakukan pekerjaan Anda, administrator Anda memberi Anda kredensi dan izin yang Anda butuhkan. Saat Anda menggunakan lebih banyak GameLift fitur Amazon 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, lihatMemecahkan masalah GameLift identitas dan akses Amazon.

Administrator layanan - Jika Anda bertanggung jawab atas sumber GameLift daya Amazon di perusahaan Anda, Anda mungkin memiliki akses penuh ke Amazon GameLift. Tugas Anda adalah menentukan GameLift fitur dan sumber daya Amazon mana yang harus diakses pengguna layanan Anda. Kemudian, Anda harus mengirimkan permintaan kepada administrator IAM Anda untuk mengubah izin pengguna layanan Anda. Tinjau informasi di halaman ini untuk memahami konsep Basic IAM. Untuk mempelajari lebih lanjut tentang bagaimana perusahaan Anda dapat menggunakan IAM dengan Amazon GameLift, lihatBagaimana Amazon GameLift bekerja dengan IAM.

Administrator IAM - Jika Anda administrator IAM, Anda mungkin ingin mempelajari detail tentang cara menulis kebijakan untuk mengelola akses ke Amazon. GameLift Untuk melihat contoh kebijakan GameLift berbasis identitas Amazon yang dapat Anda gunakan di IAM, lihat. Contoh kebijakan berbasis identitas untuk Amazon GameLift

Mengautentikasi dengan identitas

Autentikasi adalah cara Anda untuk masuk ke AWS menggunakan kredensial identitas Anda. Anda harus terautentikasi (masuk ke AWS) sebagai Pengguna root akun AWS, sebagai pengguna IAM, atau dengan mengambil peran IAM.

Anda dapat masuk ke AWS sebagai identitas terfederasi dengan menggunakan kredensial yang disediakan melalui sumber identitas. Pengguna AWS IAM Identity Center Pengguna (Pusat Identitas IAM), autentikasi Single Sign-On perusahaan Anda, dan kredensial Google atau Facebook Anda merupakan contoh identitas terfederasi. Saat Anda masuk sebagai identitas gabungan, administrator Anda sebelumnya menyiapkan federasi identitas menggunakan peran IAM. Ketika Anda mengakses AWS dengan menggunakan federasi, Anda secara tidak langsung mengambil suatu peran.

Bergantung pada jenis pengguna Anda, Anda dapat masuk ke AWS Management Console atau portal akses AWS. Untuk informasi selengkapnya tentang cara masuk ke AWS, lihat Cara masuk ke Akun AWS dalam Panduan Pengguna AWS Sign-In.

Jika Anda mengakses AWS secara terprogram, AWS memberikan Kit Pengembangan Perangkat Lunak (SDK) dan antarmuka baris perintah (CLI) untuk menandatangani permintaan Anda secara kriptografis dengan menggunakan kredensial Anda. Jika Anda tidak menggunakan peralatan AWS, Anda harus menandatangani permintaan sendiri. Untuk informasi selengkapnya tentang cara menggunakan metode yang disarankan untuk menandatangani permintaan sendiri, lihat Menandatangani permintaan API AWS dalam Panduan Pengguna IAM.

Apa pun metode autentikasi yang digunakan, Anda mungkin diminta untuk menyediakan informasi keamanan tambahan. Sebagai contoh, AWS menyarankan Anda menggunakan autentikasi multi-faktor (MFA) untuk meningkatkan keamanan akun Anda. Untuk mempelajari lebih lanjut, lihat Autentikasi multi-faktor dalam Panduan Pengguna AWS IAM Identity Center dan Menggunakan autentikasi multi-faktor (MFA) di AWS dalam Panduan Pengguna IAM.

Pengguna root Akun AWS

Ketika membuat Akun AWS, Anda memulai dengan satu identitas masuk yang memiliki akses penuh ke semua Layanan AWS dan sumber daya di akun tersebut. Identitas ini disebut pengguna root Akun AWS dan diakses dengan cara masuk menggunakan alamat email dan kata sandi yang Anda gunakan untuk membuat akun. Kami sangat menyarankan agar Anda tidak menggunakan pengguna root untuk tugas sehari-hari Anda. Lindungi kredensial pengguna root Anda dan gunakan kredensial tersebut untuk melakukan tugas yang hanya dapat dilakukan pengguna root. Untuk daftar tugas

lengkap yang mengharuskan Anda masuk sebagai pengguna root, lihat Tugas yang memerlukan kredensial pengguna root dalam Panduan Pengguna IAM.

Identitas terfederasi

Praktik terbaiknya adalah mewajibkan pengguna manusia, termasuk pengguna yang memerlukan akses administrator, untuk menggunakan federasi dengan penyedia identitas untuk mengakses Layanan AWS dengan menggunakan kredensial temporer.

Identitas terfederasi adalah pengguna dari direktori pengguna perusahaan Anda, penyedia identitas web, AWS Directory Service, direktori Pusat Identitas, atau pengguna mana pun yang mengakses Layanan AWS dengan menggunakan kredensial yang disediakan melalui sumber identitas. Ketika identitas terfederasi mengakses Akun AWS, identitas tersebut mengambil peran, dan peran ini memberikan kredensial sementara.

Untuk pengelolaan akses terpusat, sebaiknya 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 Akun AWS dan aplikasi Anda. Untuk informasi tentang Pusat Identitas IAM, lihat Apa yang dimaksud Pusat Identitas IAM? dalam Panduan Pengguna AWS IAM Identity Center.

Pengguna dan grup IAM

Pengguna IAM adalah identitas dalam Akun AWS Anda yang memiliki izin khusus untuk satu orang atau aplikasi. Jika memungkinkan, sebaiknya andalkan kredensial temporer, dan bukan membuat pengguna IAM yang memiliki kredensial jangka panjang seperti kata sandi dan kunci akses. Namun, jika Anda memiliki kasus penggunaan khusus yang memerlukan kredensial jangka panjang dengan pengguna IAM, sebaiknya rotasikan 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 kumpulan pengguna IAM. Anda tidak dapat masuk sebagai grup. Anda dapat menggunakan grup untuk menentukan izin untuk beberapa pengguna sekaligus. Grup membuat izin lebih mudah dikelola untuk sekelompok besar pengguna. Misalnya, Anda dapat memiliki grup yang bernama 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 tersebut dimaksudkan untuk dapat diambil oleh siapa pun yang membutuhkannya. Pengguna memiliki kredensial jangka panjang permanen, tetapi peran memberikan kredensial sementara. Untuk

mempelajari selengkapnya, silakan lihat Kapan harus membuat pengguna IAM (bukan peran) dalam Panduan Pengguna IAM.

Peran IAM

Peran IAM merupakan identitas dalam Akun AWS Anda yang memiliki izin khusus. Peran ini mirip dengan pengguna IAM, tetapi tidak terkait dengan orang tertentu. Anda dapat mengambil peran IAM untuk sementara dalam AWS Management Console dengan berganti peran. Anda dapat mengambil peran dengan cara memanggil operasi API AWS CLI atau AWS atau menggunakan URL kustom. Untuk informasi selengkapnya tentang metode untuk menggunakan peran, lihat Menggunakan peran IAM dalam Panduan Pengguna IAM.

Peran IAM dengan kredensial sementara berguna dalam situasi berikut:

- Akses pengguna gabungan – Untuk menetapkan izin ke sebuah identitas gabungan, Anda dapat membuat peran dan menentukan izin untuk peran tersebut. Saat identitas terfederasi diautentikasi, identitas tersebut dikaitkan dengan peran dan diberikan izin yang ditentukan oleh peran. Untuk informasi tentang peran untuk federasi, lihat Membuat peran untuk Penyedia Identitas pihak ketiga dalam Panduan Pengguna IAM. Jika Anda menggunakan Pusat Identitas IAM, Anda mengonfigurasi sekumpulan izin. Untuk mengontrol apa yang dapat diakses identitas Anda setelah identitas tersebut diautentikasi, Pusat Identitas IAM mengaitkan izin yang ditetapkan ke peran dalam IAM. Untuk informasi tentang rangkaian izin, lihat Rangkaian 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 (pengguna utama tepercaya) dengan akun berbeda untuk mengakses sumber daya yang ada di akun Anda. Peran adalah cara utama untuk memberikan akses lintas akun. Namun, pada beberapa Layanan AWS, Anda dapat menyertakan kebijakan secara langsung ke sumber daya (bukan menggunakan peran sebagai proksi). Untuk mempelajari perbedaan antara kebijakan peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat Bagaimana peran IAM berbeda dari kebijakan berbasis sumber daya dalam Panduan Pengguna IAM.
- Akses lintas layanan – Sebagian Layanan AWS menggunakan fitur di Layanan AWS lainnya. Contoh, ketika Anda melakukan panggilan dalam layanan, umumnya layanan tersebut menjalankan aplikasi di Amazon EC2 atau menyimpan objek di Amazon S3. Suatu layanan mungkin melakukan hal tersebut menggunakan izin pengguna utama panggilan, menggunakan peran layanan, atau peran terkait layanan.

- Sesi akses maju (FAS) – Ketika Anda menggunakan pengguna IAM atau peran IAM untuk melakukan tindakan di AWS, Anda akan dianggap sebagai seorang pengguna utama. Saat menggunakan beberapa layanan, Anda mungkin melakukan tindakan yang kemudian dilanjutkan oleh tindakan lain pada layanan yang berbeda. FAS menggunakan izin dari pengguna utama untuk memanggil Layanan AWS, yang dikombinasikan dengan Layanan AWS yang diminta untuk membuat permintaan ke layanan hilir. Permintaan FAS hanya diajukan saat layanan menerima permintaan yang memerlukan interaksi dengan Layanan AWS lain atau sumber daya lain untuk diselesaikan. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan ketika mengajukan permintaan FAS, lihat Meneruskan sesi akses.
- Peran IAM – Peran layanan adalah peran IAM yang diambil layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, memodifikasi, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat Membuat peran untuk mendelegasikan izin ke Layanan AWS dalam Panduan pengguna IAM.
- Peran terkait layanan – Peran terkait layanan adalah tipe peran layanan yang terkait dengan Layanan AWS. Layanan tersebut dapat mengambil peran untuk melakukan sebuah tindakan atas nama Anda. Peran terkait layanan akan muncul di Akun AWS Anda dan dimiliki oleh layanan tersebut. 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 kredensial sementara untuk aplikasi yang berjalan di instans EC2 dan mengajukan permintaan API AWS CLI atau AWS. Cara ini lebih dianjurkan daripada menyimpan kunci akses dalam instans EC2. Untuk menetapkan peran AWS ke instans EC2 dan menyediakannya bagi semua aplikasinya, Anda dapat membuat profil instans yang dilampirkan ke instans tersebut. Profil instans berisi peran dan memungkinkan program yang berjalan di instans EC2 mendapatkan kredensial sementara. Untuk informasi selengkapnya, lihat Menggunakan peran IAM untuk memberikan izin ke aplikasi yang berjalan di instans Amazon EC2 dalam Panduan Pengguna IAM.

Untuk mempelajari apakah kita harus menggunakan peran IAM atau pengguna IAM, lihat Kapan harus membuat peran IAM (bukan pengguna) dalam Panduan Pengguna IAM.

Mengelola akses menggunakan kebijakan

Anda mengendalikan akses di AWS dengan membuat kebijakan dan melampirkannya ke identitas atau sumber daya AWS. Kebijakan adalah objek di AWS yang, ketika terkait dengan identitas atau sumber daya, akan menentukan izinnya. AWS mengevaluasi kebijakan-kebijakan tersebut ketika

seorang pengguna utama (pengguna, pengguna root, atau sesi peran) mengajukan permintaan. Izin dalam kebijakan menentukan apakah permintaan diizinkan atau ditolak. Sebagian besar kebijakan disimpan di AWS sebagai dokumen JSON. Untuk informasi selengkapnya tentang struktur dan isi dokumen kebijakan JSON, silakan lihat Gambaran Umum kebijakan JSON dalam Panduan Pengguna IAM.

Administrator dapat menggunakan kebijakan JSON AWS untuk menentukan secara spesifik siapa yang memiliki akses terhadap apa. Artinya, pengguna utama manakah yang dapat melakukan tindakan pada sumber daya apa, dan dalam kondisi apa.

Secara default, pengguna dan peran tidak memiliki izin. Untuk memberikan izin kepada pengguna untuk melakukan tindakan pada sumber daya yang mereka perlukan, administrator IAM dapat membuat kebijakan IAM. Administrator kemudian dapat menambahkan kebijakan IAM ke peran, dan pengguna dapat menjalankan peran.

Kebijakan IAM mendefinisikan izin untuk suatu tindakan terlepas dari metode yang Anda gunakan untuk operasi. Sebagai contoh, anggap saja Anda memiliki kebijakan yang mengizinkan tindakan `iam:GetRole`. Pengguna dengan kebijakan tersebut dapat memperoleh informasi peran dari AWS Management Console, AWS CLI, atau API AWS.

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 pengguna dan peran, di sumber daya mana, dan dengan ketentuan apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat Membuat kebijakan IAM dalam Panduan Pengguna IAM.

Kebijakan berbasis identitas dapat dikategorikan lebih lanjut sebagai kebijakan inline atau kebijakan terkelola. Kebijakan inline disematkan langsung ke satu pengguna, grup, atau peran. Kebijakan terkelola adalah kebijakan mandiri yang dapat Anda lampirkan ke beberapa pengguna, grup, dan peran di Akun AWS Anda. Kebijakan terkelola meliputi kebijakan yang dikelola AWS dan kebijakan yang dikelola pelanggan. Untuk mempelajari cara memilih antara kebijakan terkelola atau kebijakan inline, lihat Memilih antara kebijakan terkelola 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 yang dilampiri kebijakan tersebut, kebijakan ini menentukan jenis tindakan yang dapat dilakukan oleh pengguna utama tertentu di sumber daya tersebut dan apa ketentuannya. Anda harus menentukan pengguna utama dalam kebijakan berbasis sumber daya. Pengguna utama dapat mencakup akun, pengguna, peran, pengguna gabungan, atau Layanan AWS.

Kebijakan berbasis sumber daya merupakan kebijakan inline yang terletak di layanan tersebut. Anda tidak dapat menggunakan kebijakan yang dikelola AWS dari IAM dalam kebijakan berbasis sumber daya.

Daftar kontrol akses (ACL)

Daftar kontrol akses (ACL) mengendalikan pengguna utama mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACL serupa dengan kebijakan berbasis sumber daya, meskipun tidak menggunakan format dokumen kebijakan JSON.

Amazon S3, AWS WAF, dan Amazon VPC adalah contoh layanan yang mendukung ACL. Untuk mempelajari ACL selengkapnya, silakan lihat Gambaran umum daftar kontrol akses (ACL) di Panduan Developer Layanan Penyimpanan Ringkas Amazon.

Tipe kebijakan lain

AWS mendukung jenis kebijakan tambahan yang kurang umum. Tipe-tipe kebijakan ini dapat mengatur izin maksimum yang diberikan kepada Anda berdasarkan tipe kebijakan yang lebih umum.

- Batasan izin – Batasan izin adalah fitur lanjutan di mana Anda menetapkan izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas kepada entitas IAM (pengguna 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 secara eksplisit terhadap salah satu kebijakan ini akan mengesampingkan izin tersebut. Untuk informasi selengkapnya tentang batasan izin, lihat Batasan izin untuk entitas IAM dalam Panduan Pengguna IAM.
- Kebijakan kontrol layanan (SCP) – SCP adalah kebijakan JSON yang menentukan izin maksimum untuk sebuah organisasi atau unit organisasi (OU) di AWS Organizations. AWS Organizations adalah layanan untuk mengelompokkan dan mengelola beberapa akun Akun AWS yang dimiliki bisnis Anda secara terpusat. Jika Anda mengaktifkan semua fitur di organisasi, Anda dapat

menerapkan kebijakan kontrol layanan (SCP) ke salah satu atau semua akun Anda. SCP membatasi izin untuk entitas dalam akun anggota, termasuk setiap Pengguna root akun AWS. Untuk informasi selengkapnya tentang Organisasi dan SCP, lihat Cara kerja SCP dalam Panduan Pengguna AWS Organizations.

- Kebijakan sesi – Kebijakan sesi adalah kebijakan lanjutan yang Anda teruskan sebagai parameter saat Anda membuat sesi sementara secara terprogram untuk peran atau pengguna gabungan. 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 di salah satu kebijakan ini akan membatalkan izin tersebut. Untuk informasi selengkapnya, lihat Kebijakan sesi dalam Panduan Pengguna IAM.

Berbagai jenis kebijakan

Jika beberapa jenis kebijakan diberlakukan untuk satu permintaan, izin yang dihasilkan lebih rumit untuk dipahami. Untuk mempelajari cara AWS menentukan apakah akan mengizinkan permintaan ketika ada beberapa jenis kebijakan, lihat Logika evaluasi kebijakan dalam Panduan Pengguna IAM.

Bagaimana Amazon GameLift bekerja dengan IAM

Sebelum Anda menggunakan IAM untuk mengelola akses ke Amazon GameLift, pelajari fitur IAM apa yang tersedia untuk digunakan dengan Amazon. GameLift

Fitur IAM yang dapat Anda gunakan dengan Amazon GameLift

Fitur IAM	GameLift Dukungan Amazon
Kebijakan berbasis identitas	Ya
Kebijakan berbasis sumber daya	Tidak
Tindakan kebijakan	Ya
Sumber daya kebijakan	Ya
kunci-kunci persyaratan kebijakan (spesifik layanan)	Ya
ACL	Tidak

Fitur IAM	GameLift Dukungan Amazon
ABAC (tanda dalam kebijakan)	Ya
Kredensial sementara	Ya
Izin pengguna utama	Ya
Peran layanan	Ya
Peran terkait layanan	Tidak

Untuk mendapatkan tampilan tingkat tinggi tentang cara kerja Amazon GameLift dan AWS layanan lainnya dengan sebagian besar fitur IAM, lihat AWSlayanan yang bekerja dengan IAM di Panduan Pengguna IAM.

Kebijakan berbasis identitas untuk Amazon GameLift

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 pengguna dan peran, di sumber daya mana, dan dengan ketentuan apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat Membuat kebijakan IAM dalam Panduan Pengguna IAM.

Dengan kebijakan berbasis identitas IAM, Anda dapat menentukan tindakan dan sumber daya yang diizinkan atau ditolak, serta ketentuan terkait jenis tindakan yang diizinkan atau ditolak. Anda tidak dapat menentukan pengguna utama dalam kebijakan berbasis identitas karena kebijakan ini berlaku untuk pengguna atau peran yang dilampiri kebijakan. Untuk mempelajari semua elemen yang dapat digunakan dalam kebijakan JSON, lihat Referensi elemen kebijakan JSON IAM dalam Panduan Pengguna IAM.

Contoh kebijakan berbasis identitas untuk Amazon GameLift

Untuk melihat contoh kebijakan GameLift berbasis identitas Amazon, lihat. Contoh kebijakan berbasis identitas untuk Amazon GameLift

Kebijakan berbasis sumber daya di Amazon GameLift

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 yang dilampiri kebijakan tersebut, kebijakan ini menentukan jenis tindakan yang dapat dilakukan oleh pengguna utama tertentu di sumber daya tersebut dan apa ketentuannya. Anda harus menentukan pengguna utama dalam kebijakan berbasis sumber daya. Pengguna utama dapat mencakup akun, pengguna, peran, pengguna gabungan, atau Layanan AWS.

Untuk mengaktifkan akses lintas akun, Anda dapat menentukan seluruh akun atau entitas IAM di akun lain sebagai pengguna utama dalam kebijakan berbasis sumber daya. Menambahkan pengguna utama lintas akun ke kebijakan berbasis sumber daya bagian dari membangun hubungan kepercayaan. Ketika pengguna utama dan sumber daya berada di Akun AWS yang berbeda, administrator IAM di akun tepercaya juga harus memberikan izin kepada entitas pengguna utama (pengguna atau peran) untuk mengakses sumber daya. Izin diberikan dengan melampirkan kebijakan berbasis identitas ke entitas tersebut. Namun, jika kebijakan berbasis sumber daya memberikan akses kepada pengguna utama dalam akun yang sama, kebijakan berbasis identitas lainnya tidak diperlukan. Untuk informasi selengkapnya, lihat Perbedaan peran IAM dengan kebijakan berbasis sumber daya di Panduan Pengguna IAM.

Tindakan kebijakan untuk Amazon GameLift

Mendukung tindakan kebijakan	Ya

Administrator dapat menggunakan kebijakan JSON AWS untuk menentukan siapa yang memiliki akses ke apa. Yaitu, pengguna utama mana yang dapat melakukan tindakan pada sumber daya apa, 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 seperti operasi API AWS 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.

Menyertakan tindakan dalam suatu kebijakan untuk memberikan izin melakukan operasi terkait.

Untuk daftar GameLift tindakan Amazon, lihat Tindakan yang ditentukan oleh Amazon GameLift di Referensi Otorisasi Layanan.

Tindakan kebijakan di Amazon GameLift menggunakan awalan berikut sebelum tindakan:

```
gamelift
```

Untuk menetapkan secara spesifik beberapa tindakan dalam satu pernyataan, pisahkan tindakan-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 kebijakan GameLift berbasis identitas Amazon, lihat. Contoh kebijakan berbasis identitas untuk Amazon GameLift

Sumber daya kebijakan untuk Amazon GameLift

Mendukung sumber daya kebijakan	Ya

Administrator dapat menggunakan kebijakan JSON AWS untuk menentukan siapa yang memiliki akses ke apa. Yaitu, pengguna utama mana yang dapat melakukan tindakan pada sumber daya apa, dan dalam kondisi apa.

Elemen kebijakan JSON `Resource` menentukan objek atau beberapa 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 mengindikasikan bahwa pernyataan tersebut berlaku untuk semua sumber daya.

```
"Resource": "*"
```

Untuk daftar jenis GameLift sumber daya Amazon dan ARNnya, lihat Sumber daya yang ditentukan oleh Amazon GameLift di Referensi Otorisasi Layanan. Untuk mempelajari tindakan mana yang dapat Anda tentukan ARN dari setiap sumber daya, lihat Tindakan yang ditentukan oleh Amazon. GameLift

Beberapa GameLift sumber daya Amazon memiliki nilai ARN, yang memungkinkan sumber daya untuk mengelola aksesnya menggunakan kebijakan IAM. Sumber daya GameLift armada Amazon memiliki ARN dengan sintaks berikut:

```
arn:${Partition}:gamelift:${Region}:${Account}:fleet/${FleetId}
```

Untuk informasi selengkapnya tentang format ARN, lihat Amazon Resource Name (ARN) 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 kebijakan GameLift berbasis identitas Amazon, lihat. Contoh kebijakan berbasis identitas untuk Amazon GameLift

Kunci kondisi kebijakan untuk Amazon GameLift

Mendukung kunci kondisi kebijakan spesifik layanan	Ya

Administrator dapat menggunakan kebijakan JSON AWS untuk menentukan siapa yang memiliki akses ke apa. Yaitu, pengguna utama mana yang dapat melakukan tindakan pada sumber daya apa, dan dalam kondisi apa.

Elemen `Condition` (atau blok `Condition`) memungkinkan Anda menentukan kondisi di mana suatu pernyataan akan diterapkan. Elemen `Condition` bersifat opsional. Anda dapat membuat ekspresi kondisional 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 satu pernyataan, atau beberapa kunci dalam satu elemen `Condition`, AWS akan mengevaluasinya dengan menggunakan operasi `AND` logis. Jika Anda menentukan beberapa niilai untuk satu kunci persyaratan, AWS akan mengevaluasi syarat tersebut menggunakan operasi `OR` yang 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, silakan lihat Elemen kebijakan IAM: variabel dan tanda di Panduan Pengguna IAM.

AWS mendukung kunci kondisi global dan kunci kondisi spesifik layanan. Untuk melihat semua kunci kondisi global AWS, lihat kunci konteks kondisi global AWS dalam Panduan Pengguna IAM.

Untuk daftar kunci GameLift kondisi Amazon, lihat Kunci kondisi untuk Amazon GameLift di Referensi Otorisasi Layanan. Untuk mempelajari tindakan dan sumber daya yang dapat Anda gunakan kunci kondisi, lihat Tindakan yang ditentukan oleh Amazon GameLift.

Untuk melihat contoh kebijakan GameLift berbasis identitas Amazon, lihat. Contoh kebijakan berbasis identitas untuk Amazon GameLift

ACL di Amazon GameLift

Mendukung ACL	Tidak

Daftar kontrol akses (ACL) mengontrol pengguna utama (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACL sama dengan kebijakan berbasis sumber daya, meskipun tidak menggunakan format dokumen kebijakan JSON.

ABAC dengan Amazon GameLift

Mendukung ABAC (tanda dalam kebijakan)	Ya

Kontrol akses berbasis atribut (ABAC) adalah strategi otorisasi yang mendefinisikan izin berdasarkan atribut. Di AWS, atribut ini disebut tag. Anda dapat melampirkan tanda ke entitas IAM (pengguna atau peran) dan ke banyak sumber daya AWS. Pemberian tanda ke entitas dan sumber daya adalah langkah pertama dari ABAC. Kemudian rancanglah kebijakan ABAC untuk mengizinkan operasi-operasi ketika tanda milik pengguna utama cocok dengan tanda yang ada di sumber daya yang ingin diakses.

ABAC sangat berguna di lingkungan yang berkembang dengan cepat dan berguna dalam situasi di mana pengelolaan kebijakan menjadi rumit.

Untuk mengendalikan akses berdasarkan tag, berikan informasi tentang tanda di elemen syarat dari sebuh kebijakan dengan menggunakan kunci-kunci persyaratan `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 hanya untuk beberapa jenis sumber daya, nilainya adalah Parsial.

Untuk informasi selengkapnya tentang ABAC, lihat Apa itu ABAC? di Panduan Pengguna IAM. Untuk melihat tutorial terkait langkah-langkah penyiapan ABAC, lihat Menggunakan kontrol akses berbasis atribut (ABAC) di Panduan Pengguna IAM.

Untuk contoh kebijakan berbasis identitas yang membatasi akses ke sumber daya berdasarkan tag pada sumber daya tersebut, lihat. Lihat GameLift armada Amazon berdasarkan tag

Menggunakan kredensyal sementara dengan Amazon GameLift

Mendukung kredensial sementara	Ya

Beberapa Layanan AWS tidak berfungsi saat Anda masuk menggunakan kredensial sementara. Sebagai informasi tambahan, termasuk tentang Layanan AWS mana saja yang berfungsi dengan kredensial sementara, lihat Layanan AWS yang berfungsi dengan IAM di Panduan Pengguna IAM.

Anda menggunakan kredensial sementara jika Anda masuk ke AWS Management Console menggunakan metode apa pun kecuali nama pengguna dan kata sandi. Misalnya, ketika Anda mengakses AWS dengan menggunakan tautan masuk tunggal (SSO) milik perusahaan Anda, proses itu secara otomatis akan membuat kredensial temporer. Anda juga akan membuat kredensial sementara secara otomatis saat masuk ke konsol sebagai pengguna dan kemudian beralih peran. Untuk informasi selengkapnya tentang cara beralih peran, lihat Beralih peran (konsol) di Panduan Pengguna IAM.

Anda dapat membuat kredensial sementara secara manual menggunakan AWS CLI atau AWS API. Anda kemudian dapat menggunakan kredensial sementara untuk mengakses AWS. AWS menyarankan Anda membuat kredensial sementara secara dinamis, alih-alih menggunakan kunci akses jangka panjang. Untuk informasi selengkapnya, lihat Kredensial keamanan sementara di IAM.

Izin utama lintas layanan untuk Amazon GameLift

Mendukung sesi akses maju (FAS)	Ya

Jika menggunakan pengguna IAM atau peran IAM untuk melakukan tindakan di AWS, Anda akan dianggap sebagai pengguna utama. Jika menggunakan beberapa layanan, Anda mungkin melakukan tindakan yang kemudian dilanjutkan oleh tindakan lain di layanan yang berbeda. FAS menggunakan izin dari pengguna utama untuk memanggil Layanan AWS, yang dikombinasikan dengan Layanan AWS yang diminta untuk membuat permintaan ke layanan hilir. Permintaan FAS hanya diajukan saat layanan menerima permintaan yang memerlukan interaksi dengan Layanan AWS lain atau sumber daya lain untuk diselesaikan. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan ketika mengajukan permintaan FAS, lihat Meneruskan sesi akses.

Peran layanan untuk Amazon GameLift

Mendukung peran layanan	Ya

Peran layanan adalah peran IAM yang diambil oleh layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, mengubah, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat Membuat peran untuk mendelegasikan izin ke Layanan AWS dalam Panduan pengguna IAM.

> **Warning**
>
> Mengubah izin untuk peran layanan dapat merusak GameLift fungsionalitas Amazon. Edit peran layanan hanya jika Amazon GameLift memberikan panduan untuk melakukannya.

Izinkan server game yang GameLift dihosting Amazon Anda mengakses AWS sumber daya lain, seperti AWS Lambda fungsi atau database Amazon DynamoDB. Karena server game di-host di armada yang GameLift dikelola Amazon, Anda memerlukan peran layanan yang memberi Amazon akses GameLift terbatas ke AWS sumber daya lainnya. Untuk informasi selengkapnya, lihat Berkomunikasi dengan sumber daya AWS lain dari armada.

Peran terkait layanan untuk Amazon GameLift

Mendukung peran terkait layanan	Tidak

Peran tertaut layanan adalah jenis peran layanan yang terkait dengan Layanan AWS. Layanan ini dapat menjalankan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan akan muncul di Akun AWS Anda dan dimiliki oleh layanan tersebut. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran terkait 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

Secara default, pengguna dan peran tidak memiliki izin untuk membuat atau memodifikasi GameLift sumber daya Amazon. Pengguan dan peran tersebut juga tidak dapat melakukan tugas dengan menggunakan AWS Management Console, AWS Command Line Interface (AWS CLI), atau API AWS. Untuk memberikan izin kepada pengguna untuk melakukan tindakan pada sumber daya yang

mereka perlukan, administrator IAM dapat membuat kebijakan IAM. Administrator kemudian dapat menambahkan kebijakan IAM ke peran, dan pengguna dapat menjalankan peran.

Untuk mempelajari cara membuat kebijakan berbasis identitas IAM menggunakan contoh dokumen kebijakan JSON ini, lihat Membuat kebijakan IAM dalam Panduan Pengguna IAM.

Untuk detail tentang tindakan dan jenis sumber daya yang ditentukan oleh Amazon GameLift, termasuk format ARN untuk setiap jenis sumber daya, lihat Kunci tindakan, sumber daya, dan kondisi untuk Amazon GameLift di Referensi Otorisasi Layanan.

Topik

- Praktik terbaik kebijakan
- Menggunakan GameLift konsol Amazon
- Izinkan pengguna melihat izin mereka sendiri
- Izinkan akses pemain untuk sesi game
- Izinkan akses ke satu GameLift antrian Amazon
- Lihat GameLift armada Amazon berdasarkan tag
- Mengakses file build game di Amazon S3

Praktik terbaik kebijakan

Kebijakan berbasis identitas menentukan apakah seseorang dapat membuat, mengakses, atau menghapus GameLift sumber daya Amazon di akun Anda. Tindakan ini dikenai biaya untuk Akun AWS Anda. Ketika Anda membuat atau mengedit kebijakan berbasis identitas, ikuti panduan dan rekomendasi ini:

- Mulai menggunakan kebijakan yang dikelola AWS dan beralih ke izin dengan hak akses paling rendah – Untuk mulai memberikan izin kepada pengguna dan beban kerja Anda, gunakan kebijakan yang dikelola AWS yang memberikan izin untuk banyak kasus penggunaan umum. Kebijakan ini ada di Akun AWS Anda. Sebaiknya Anda mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola pelanggan AWS yang khusus untuk kasus penggunaan Anda. Untuk informasi selengkapnya, lihat kebijakan yang dikelola AWS atau kebijakan yang dikelola AWS untuk fungsi pekerjaan di 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 melakukan ini dengan menentukan tindakan yang dapat diambil pada sumber daya tertentu dalam kondisi

tertentu, juga dikenal sebagai izin hak akses paling rendah. Untuk informasi selengkapnya tentang cara menggunakan IAM untuk menerapkan izin, lihat Kebijakan dan izin di IAM di Panduan Pengguna IAM.

- Gunakan kondisi dalam kebijakan IAM untuk membatasi akses lebih lanjut – Anda dapat menambahkan kondisi ke kebijakan Anda untuk membatasi akses ke tindakan dan sumber daya. Misalnya, Anda dapat menulis syarat kebijakan untuk menentukan bahwa semua pengajuan harus dikirim menggunakan SSL. Anda juga dapat menggunakan kondisi untuk memberi akses ke tindakan layanan jika digunakan melalui Layanan AWS yang spesifk, seperti AWS CloudFormation. Untuk informasi selengkapnya, lihat Elemen kebijakan JSON IAM: Syarat di Panduan Pengguna IAM.
- Menggunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda guna memastikan izin yang aman dan berfungsi – 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 Analizer Akses IAM di Panduan Pengguna IAM.
- Wajibkan autentikasi multi-faktor (MFA) – Jika Anda memiliki skenario yang mengharuskan pengguna IAM atau pengguna root di Akun AWS Anda, aktifkan MFA untuk keamanan tambahan. Untuk mewajibkan MFA saat operasi API dipanggil, tambahkan kondisi MFA pada kebijakan Anda. Untuk informasi selengkapnya, lihat Mengonfigurasi akses API yang dilindungi MFA di Panduan Pengguna IAM.

Untuk informasi selengkapnya tentang praktik terbaik dalam IAM, lihat Praktik terbaik keamanan di IAM di Panduan Pengguna IAM.

Menggunakan GameLift konsol Amazon

Untuk mengakses GameLift konsol Amazon, Anda harus memiliki set izin minimum. Izin ini harus memungkinkan Anda untuk membuat daftar dan melihat detail tentang GameLift sumber daya Amazon di AndaAkun 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 GameLift konsol Amazon, tambahkan izin ke pengguna dan grup dengan sintaks dalam contoh berikut dan di. Contoh izin administrator Untuk informasi selengkapnya, lihat Mengelola izin pengguna untuk Amazon GameLift.

Pengguna yang bekerja dengan Amazon GameLift 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 fitur GameLift konsol Amazon, lihat sintaks izin untuk administrator di. Contoh izin administrator

Izinkan pengguna melihat izin mereka sendiri

Contoh ini menunjukkan cara membuat kebijakan yang mengizinkan para pengguna IAM melihat kebijakan inline dan terkelola yang dilampirkan ke identitas pengguna mereka. Kebijakan ini mencakup izin untuk menyelesaikan tindakan pada konsol atau secara programatis menggunakan API AWS CLI atau 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, lihatContoh izin pengguna pemain.

Izinkan akses ke satu GameLift antrian Amazon

Contoh berikut memberi pengguna akses ke GameLift antrian Amazon tertentu.

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"
      }
```

```
    ]
}
```

Lihat GameLift armada Amazon berdasarkan tag

Anda dapat menggunakan kondisi dalam kebijakan berbasis identitas untuk mengontrol akses ke GameLift sumber daya Amazon 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, unggah file build ke Amazon S3. GameLift Agar Amazon dapat 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 selengkapnya tentang mengunggah file GameLift game Amazon, lihatUnggah build server khusus ke Amazon GameLift.

Memecahkan masalah GameLift identitas dan akses Amazon

Gunakan informasi berikut untuk membantu Anda mendiagnosis dan memperbaiki masalah umum yang mungkin Anda temui saat bekerja dengan Amazon GameLift dan AWS Identity and Access Management (IAM).

Topik

- Saya tidak berwenang untuk melakukan tindakan di Amazon GameLift
- Saya tidak berwenang untuk melakukan iam: PassRole
- Saya ingin mengizinkan orang di luar saya Akun AWS untuk mengakses GameLift sumber daya Amazon saya

Saya tidak berwenang untuk melakukan tindakan di Amazon GameLift

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 bahwa Anda tidak diizinkan untuk melakukan `iam:PassRole` tindakan, kebijakan Anda harus diperbarui agar Anda dapat meneruskan peran ke Amazon GameLift.

Sebagian Layanan AWS mengizinkan Anda untuk memberikan peran yang sudah ada ke layanan tersebut alih-alih membuat peran layanan baru atau peran terkait-layanan. Untuk melakukan tindakan tersebut, Anda harus memiliki izin untuk memberikan peran pada layanan tersebut.

Contoh kesalahan berikut terjadi ketika pengguna IAM bernama `marymajor` mencoba menggunakan konsol untuk melakukan tindakan di Amazon GameLift. Namun, tindakan tersebut memerlukan layanan untuk mendapatkan 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 membutuhkan bantuan, hubungi administrator AWS Anda. Administrator Anda adalah orang yang memberi Anda kredensial masuk.

Saya ingin mengizinkan orang di luar saya Akun AWS untuk mengakses GameLift sumber daya Amazon saya

Anda dapat membuat peran yang dapat digunakan pengguna di akun lain atau pengguna 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 (ACL), Anda dapat menggunakan kebijakan tersebut untuk memberi pengguna akses ke sumber daya Anda.

Untuk mempelajari selengkapnya, periksa hal berikut:

- Untuk mengetahui apakah Amazon GameLift mendukung fitur-fitur ini, lihatBagaimana Amazon GameLift bekerja dengan IAM.

- Untuk mempelajari cara memberikan akses ke sumber daya di seluruh Akun AWS yang Anda miliki, lihat Menyediakan akses ke pengguna IAM di Akun AWS lainnya yang Anda miliki dalam Panduan Pengguna IAM.
- Untuk mempelajari cara memberikan akses ke sumber daya Anda ke pihak ketiga Akun AWS, lihat Menyediakan akses ke akun Akun AWS yang dimiliki oleh pihak ketiga dalam Panduan Pengguna IAM.
- Untuk mempelajari cara memberikan akses melalui federasi identitas, lihat Menyediakan akses ke pengguna terautentikasi eksternal (gabungan identitas) dalam Panduan Pengguna IAM.
- Untuk mempelajari perbedaan antara penggunaan peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat Perbedaan antara peran IAM dan kebijakan berbasis sumber daya di Panduan Pengguna IAM.

Pencatatan dan pemantauan dengan Amazon GameLift

Pemantauan adalah bagian penting dalam menjaga keandalan, ketersediaan, dan kinerja Amazon GameLift dan AWS solusi Anda. Anda harus mengumpulkan data pemantauan dari semua bagian solusi AWS sehingga Anda dapat melakukan debug kegagalan multititik secara lebih mudah jika terjadi kegagalan.

AWSdan Amazon GameLift menyediakan beberapa alat untuk memantau sumber daya hosting game Anda dan menanggapi potensi insiden.

Alarm Amazon CloudWatch

Menggunakan alarm Amazon CloudWatch, Anda melihat satu metrik selama periode waktu yang Anda tentukan. Jika metrik melebihi ambang batas tertentu, notifikasi dikirim ke topik Amazon SNS atau kebijakan Auto Scaling AWS. CloudWatchalarm dipicu ketika negara mereka berubah dan dipertahankan untuk sejumlah periode tertentu, bukan dengan berada dalam keadaan tertentu. Untuk informasi selengkapnya, lihat Pantau Amazon GameLift dengan Amazon CloudWatch.

Log AWS CloudTrail

CloudTrailmenyediakan catatan tindakan yang diambil oleh pengguna, peran, atau AWS layanan di AmazonGameLift. Menggunakan informasi yang dikumpulkan olehCloudTrail, Anda dapat menentukan permintaan yang dibuat untuk AmazonGameLift, alamat IP dari mana permintaan dibuat, yang membuat permintaan, ketika dibuat, dan rincian tambahan. Untuk informasi selengkapnya, lihat Mencatat panggilan GameLift API Amazon dengan AWS CloudTrail.

Validasi kepatuhan untuk Amazon GameLift

Amazon GameLift tidak dalam lingkup program AWS kepatuhan apa pun.

Untuk mempelajari apakah Layanan AWS berada dalam lingkup program kepatuhan khusus, lihat Layanan AWS di Scope oleh Program Program Kepatuhan yang Anda minati. Untuk informasi umum, lihat Program Kepatuhan 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 berdasarkan sensitivitas data Anda, tujuan kepatuhan perusahaan Anda, serta hukum dan peraturan yang berlaku. AWS menyediakan sumber daya berikut untuk membantu kepatuhan:

- Panduan Mulai Cepat Keamanan dan Kepatuhan – Panduan deployment ini membahas pertimbangan arsitektur dan menyediakan langkah-langkah untuk melakukan deployment lingkungan dasar di AWS yang menjadi fokus keamanan dan kepatuhan.
- Merancang Keamanan dan Kepatuhan HIPAA di Amazon Web Services – Laporan resmi ini menjelaskan cara perusahaan dapat menggunakan AWS untuk membuat aplikasi yang memenuhi syarat HIPAA.

> **Note**
>
> Tidak semua Layanan AWS memenuhi syarat HIPAA. Untuk informasi selengkapnya, lihat Referensi Layanan yang Memenuhi Syarat HIPAA.

- Sumber Daya Kepatuhan AWS – Kumpulan buku kerja dan panduan ini mungkin berlaku untuk industri dan lokasi Anda.
- Panduan Kepatuhan Pelanggan AWS – Pahami model tanggung jawab bersama melalui lensa kepatuhan. Panduan ini merangkum praktik terbaik untuk mengamankan Layanan AWS dan memetakan panduan kontrol keamanan di banyak kerangka kerja (termasuk National Institute of Standards and Technology (NIST), Payment Card Industry Security Standards Council (PCI), dan International Organization for Standardization (ISO)).
- Mengevaluasi Sumber Daya dengan Aturan dalam Panduan Developer AWS Config – Layanan AWS Config menilai seberapa baik konfigurasi sumber daya Anda dalam mematuhi praktik-praktik internal, pedoman industri, dan regulasi internal.

- AWS Security Hub – Layanan AWS ini memberikan pandangan komprehensif tentang status keamanan Anda di dalam AWS. Security Hub menggunakan kontrol keamanan untuk mengevaluasi sumber daya AWS Anda dan memeriksa kepatuhan Anda terhadap standar industri keamanan dan praktik terbaik. Untuk daftar layanan dan kontrol yang didukung, lihat Referensi kontrol Security Hub.
- AWS Audit Manager – Layanan AWS ini akan membantu Anda untuk terus-menerus mengaudit penggunaan AWS untuk menyederhanakan cara Anda mengelola risiko dan kepatuhan terhadap regulasi dan standar industri.

Ketahanan di Amazon GameLift

Jika Anda menggunakan Amazon GameLift FleetiQ sebagai fitur mandiri dengan Amazon EC2, lihat Keamanan di Amazon EC2 di Panduan Pengguna Amazon EC2 untuk Instans Linux.

Infrastruktur global AWS dibangun di sekitar Wilayah AWS dan Availability Zone. AWS Wilayah menyediakan beberapa Availability Zone yang terpisah secara fisik dan terisolasi, yang terhubung dengan jaringan berlatensi rendah, throughput yang tinggi, dan sangat redundan. Dengan Availability Zone, Anda dapat mendesain dan mengoperasikan aplikasi dan basis data yang secara otomatis mengalami kegagalan di antara zona tanpa gangguan. Availability Zone lebih tersedia, memiliki toleransi kesalahan, dan dapat diskalakan dibandingkan dengan satu atau beberapa infrastruktur pusat data tradisional.

Untuk informasi selengkapnya tentang Wilayah AWS dan Availability Zone, lihat Infrastruktur global AWS.

Selain infrastruktur AWS global, Amazon GameLift menawarkan fitur-fitur berikut untuk membantu mendukung kebutuhan ketahanan data Anda:

- Antrian multi-wilayah — Antrian sesi GameLift game Amazon digunakan untuk menempatkan sesi game 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 Desain antrean sesi game.
- Penskalaan kapasitas otomatis - Pertahankan kesehatan dan ketersediaan sumber daya hosting Anda dengan menggunakan alat GameLift penskalaan Amazon. Alat-alat ini menyediakan berbagai opsi yang memungkinkan Anda menyesuaikan kapasitas armada agar sesuai dengan kebutuhan game dan pemain Anda. Untuk informasi lebih lanjut tentang penskalaan, lihat Penskalaan kapasitas GameLift hosting Amazon.

- Distribusi di seluruh instans — Amazon GameLift mendistribusikan lalu lintas masuk di beberapa instans, 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 ke Amazon GameLift 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

Jika Anda menggunakan Amazon GameLift FleetiQ sebagai fitur mandiri dengan Amazon EC2, lihat Keamanan di Amazon EC2 di Panduan Pengguna Amazon EC2 untuk Instans Linux.

Sebagai layanan terkelola, Amazon GameLift dilindungi oleh prosedur keamanan jaringan AWS global yang dijelaskan dalam whitepaper Amazon Web Services: Ikhtisar proses keamanan.

Anda menggunakan panggilan API yang AWS dipublikasikan untuk mengakses Amazon GameLift melalui jaringan. Klien harus mendukung Keamanan Lapisan Pengangkutan (TLS) 1.2 atau versi yang lebih baru. Kami merekomendasikan TLS 1.3 atau yang lebih baru. Klien juga harus mendukung suite cipher 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 access key ID dan secret access key yang terkait dengan principal IAM. Atau Anda bisa menggunakan AWS Security Token Service (AWS STS) untuk membuat kredensial keamanan sementara guna menandatangani permintaan.

GameLiftLayanan Amazon menempatkan semua armada ke awan pribadi virtual Amazon (VPC) sehingga setiap armada ada di area yang terisolasi secara logis di Cloud. AWS Anda dapat menggunakan GameLift kebijakan Amazon untuk mengontrol akses dari titik akhir VPC tertentu atau VPC tertentu. Secara efektif, ini mengisolasi akses jaringan ke GameLift sumber daya Amazon tertentu 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

Jika Anda menggunakan Amazon GameLift FleetiQ sebagai fitur mandiri dengan Amazon EC2, lihat Keamanan di Amazon EC2 di Panduan Pengguna Amazon EC2 untuk Instans Linux.

Konfigurasi dan kontrol IT merupakan tanggung jawab bersama antara AWS dan Anda, pelanggan kami. Untuk informasi lebih lanjut, lihat model tanggung jawab bersama AWS. AWS menangani tugas-tugas keamanan dasar seperti sistem operasi tamu (OS) dan patching basis data, 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 AmazonGameLift:

- Pelanggan bertanggung jawab atas pengelolaan perangkat lunak yang diterapkan ke GameLift instans Amazon 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 AmazonGameLift, buat armada baru untuknya, dan alihkan 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 berkala dengan versi SDK terbaru, termasuk SDK, Amazon GameLift Server AWS SDK, dan Amazon GameLift Client SDK untuk Server Realtime.

Praktik terbaik keamanan untuk Amazon GameLift

Jika Anda menggunakan Amazon GameLift FleetiQ sebagai fitur mandiri dengan Amazon EC2, lihat Keamanan di Amazon EC2 di Panduan Pengguna Amazon EC2 untuk Instans Linux.

Amazon GameLift menyediakan sejumlah fitur keamanan untuk dipertimbangkan saat 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.

Untuk informasi selengkapnya tentang bagaimana Anda dapat membuat penggunaan Amazon GameLift lebih aman, lihat pilar AWS Well-Architected Tool Keamanan. .

Panduan GameLift referensi Amazon

Bagian ini berisi dokumentasi referensi untuk menggunakan AmazonGameLift.

Topik

- Referensi API GameLift layanan Amazon (AWSSDK)
- GameLiftReferensi Amazon Realtime Server
- Referensi SDK GameLift server Amazon
- Acara penempatan sesi game

Referensi API GameLift layanan Amazon (AWSSDK)

Topik ini menyediakan daftar operasi API berbasis tugas untuk digunakan dengan solusi hosting GameLift terkelola Amazon, termasuk hosting untuk server game khusus dan Server Realtime. Operasi ini dikemas ke dalam AWS SDK di namespace. `aws.gamelift` Unduh AWS SDK atau lihat dokumentasi referensi Amazon GameLift API.

API mencakup dua set operasi untuk hosting game terkelola:

- Menyiapkan dan mengelola sumber daya GameLift hosting Amazon
- Memulai sesi game dan bergabung dengan pemain

Amazon GameLift Service API juga berisi operasi untuk digunakan dengan GameLift alat dan solusi Amazon lainnya. Untuk daftar API FleetIQ, lihat Tindakan API FleetIQ. Untuk daftar FlexMatch API untuk perjodohan, lihat tindakan FlexMatchAPI.

Menyiapkan dan mengelola sumber daya GameLift hosting Amazon

Hubungi operasi ini untuk mengonfigurasi sumber daya hosting untuk server game Anda, menskalakan kapasitas untuk memenuhi permintaan pemain, mengakses performa dan metrik pemanfaatan, dan banyak lagi. Operasi API ini digunakan dengan server game yang di-host di AmazonGameLift, termasuk Server Realtime. Anda dapat menggunakan GameLiftkonsol Amazon untuk sebagian besar tugas pengelolaan sumber daya, atau Anda dapat melakukan panggilan ke layanan menggunakan alat AWS Command Line Interface (AWS CLI) atau AWS SDK.

Siapkan server game untuk deployment

Unggah dan konfigurasikan 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 Amazon GameLift baru. Operasi ini, yang hanya tersedia sebagai perintah AWS CLI, adalah metode yang paling umum untuk mengunggah server pembuatan game.
- CreateBuild— Buat build baru menggunakan file yang disimpan dalam bucket Amazon S3.
- ListBuilds- Dapatkan daftar semua build yang diunggah ke GameLift wilayah Amazon.
- DescribeBuild- Ambil informasi yang terkait dengan build.
- UpdateBuild- Ubah metadata build, termasuk nama dan versi build.
- DeleteBuild- Hapus build dari AmazonGameLift.

Mengelola skrip konfigurasi Server Realtime

- CreateScript- Unggah JavaScript file dan buat sumber daya GameLift skrip Amazon baru.
- ListScripts- Dapatkan daftar semua skrip Realtime yang diunggah ke GameLift wilayah Amazon.
- DescribeScript- Ambil informasi yang terkait dengan skrip Realtime.
- UpdateScript- Ubah metadata skrip dan unggah konten skrip yang direvisi.
- DeleteScript- Hapus skrip Realtime dari AmazonGameLift.

Mengatur sumber daya komputasi untuk hosting

Mengonfigurasi sumber daya hosting dan men-deploy sumber daya dengan build server game Anda atau konfigurasi skrip Realtime.

Membuat dan mengelola armada

- CreateFleet- Konfigurasikan dan terapkan GameLift armada Amazon baru sumber daya komputasi 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 GameLift wilayah Amazon.

- DeleteFleet- Hentikan 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 melihat status saat ini dari setiap lokasi.
  - DeleteFleetLocations- Hapus lokasi terpencil dari armada yang mendukung beberapa lokasi.
- Melihat/memperbarui konfigurasi armada.
  - DescribeFleetAttributes/UpdateFleetAttributes— Melihat atau mengubah metadata armada dan pengaturan 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.

Mengelola kapasitas armada

- DescribeEC2 InstanceLimits - Ambil jumlah maksimum instance 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 penskalaan otomatis:
  - PutScalingPolicy- Aktifkan penskalaan otomatis berbasis target atau buat kebijakan penskalaan otomatis khusus, atau perbarui kebijakan yang ada.
  - DescribeScalingPolicies- Ambil kebijakan penskalaan otomatis yang ada.
  - DeleteScalingPolicy- Hapus kebijakan penskalaan otomatis dan hentikan agar tidak memengaruhi kapasitas armada.
  - StartFleetActions- Mulai ulang kebijakan penskalaan otomatis armada.
  - StopFleetActions- Tangguhkan kebijakan penskalaan otomatis armada.

Pantau aktivitas armada.

- DescribeFleetUtilization- Ambil statistik 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 game, termasuk waktu berjalan game dan jumlah pemain saat ini.

Mengatur antrean untuk penempatan sesi game yang optimal

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 penempatan sesi game.
- DescribeGameSessionQueues- Ambil antrean sesi game yang ditentukan di wilayah Amazon. GameLift
- UpdateGameSessionQueue- Ubah konfigurasi antrian sesi game.
- 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 didefinisikan di wilayah AmazonGameLift.
- DescribeAlias- Ambil informasi tentang alias yang ada.
- UpdateAlias- Ubah pengaturan untuk alias, seperti mengalihkannya dari satu armada ke armada lainnya.
- DeleteAlias- Hapus alias dari wilayah tersebut.
- ResolveAlias- Dapatkan ID armada yang menunjuk alias tertentu.

Mengakses instans hosting

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- Meminta kredensi 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 GameLift hosting Amazon Anda dan sumber daya lainnyaAWS.

- CreateVpcPeeringAuthorization- Otorisasi koneksi peering ke salah satu VPC Anda.
- DescribeVpcPeeringAuthorizations- Ambil otorisasi koneksi peering valid.
- DeleteVpcPeeringAuthorization- Hapus otorisasi koneksi peering.
- CreateVpcPeeringConnection- Buat koneksi peering antara VPC untuk GameLift armada Amazon dan salah satu VPC Anda.
- DescribeVpcPeeringConnections- Ambil informasi tentang koneksi peering VPC yang aktif atau tertunda dengan armada Amazon. GameLift
- DeleteVpcPeeringConnection- Hapus koneksi peering VPC dengan armada AmazonGameLift.

Memulai sesi game dan bergabung dengan pemain

Hubungi operasi ini dari layanan client game Anda untuk memulai sesi game baru, dapatkan informasi tentang sesi game yang ada, dan bergabunglah dengan pemain ke sesi game. Operasi ini untuk digunakan dengan server game khusus yang di-host di AmazonGameLift. Jika Anda menggunakan Realtime Servers, kelola sesi game menggunakan Referensi API klien Server Realtime (C #).

- Mulai sesi permainan baru untuk satu atau lebih pemain.
  - StartGameSessionPlacement- Minta Amazon GameLift 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 antrean sesi game untuk melacak ketersediaan

hosting di beberapa wilayah, dan menggunakan algoritme FleetIQ 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 game, alih-alih 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 untuk 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 game tertentu, termasuk lamanya waktu aktif dan jumlah pemain saat ini.
  - DescribeGameSessionDetails- Ambil metadata, termasuk pengaturan perlindungan sesi game, untuk satu atau beberapa sesi game.
  - 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.

GameLiftReferensi Amazon Realtime Server

Bagian ini berisi dokumentasi referensi untuk Amazon GameLift Realtime Server SDK. Ini termasuk API Realtime Client serta panduan untuk mengonfigurasi skrip Realtime Servers Anda.

Topik

- Referensi API klien Server Realtime (C #)
- Referensi skrip Amazon GameLift Realtime Server

Referensi API klien Server Realtime (C #)

Gunakan API Klien Realtime untuk mempersiapkan klien game multipemain Anda untuk digunakan dengan Amazon GameLift Realtime Server. Untuk informasi lebih lanjut tentang proses integrasi, lihat Siapkan server Realtime. API Client berisi serangkaian panggilan API sinkron dan callback asinkron yang memungkinkan client game terhubung ke server Realtime dan bertukar pesan dan data dengan client game lainnya melalui server.

API ini ditentukan di pustaka berikut:

Client.cs

- Tindakan Sinkron
- Callback Asinkron
- Jenis Data

Cara menyiapkan API klien Realtime

1. Unduh SDK klien Amazon GameLift Realtime.
2. Buat 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 membuat pustaka SDK, pulihkan NuGet paket dan buat solusinya.
3. Tambahkan library Klien Realtime ke proyek klien game Anda.

Referensi API klien Server Realtime (C #): Tindakan

Referensi API Klien C # Realtime ini dapat membantu Anda mempersiapkan game multipemain untuk digunakan dengan Server Realtime yang digunakan di armada Amazon. GameLift Untuk informasi detail tentang proses integrasi, lihat Siapkan server Realtime.

- Tindakan Sinkron
- Callback Asinkron
- Jenis 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 koneksi client/server. Anda dapat memilih untuk memanggil Client() tanpa parameter ini; namun, pendekatan ini secara default menghasilkan koneksi yang tidak aman.

Tipe: ClientConfiguration

Wajib: Tidak

Nilai kembali

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

AWSSDK Amazon GameLift API StartGameSessionPlacement, CreateGameSession, atau. DescribeGameSessions

> **Note**
>
> Jika server Realtime berjalan pada armada dengan sertifikat TLS, Anda harus menggunakan nama DNS.

Jenis: String

Wajib: Ya

remoteTcpPort

Nomor port untuk koneksi TCP yang ditetapkan ke sesi game. Informasi ini ditentukan dalam sebuah `GameSession` objek, yang dikembalikan dalam menanggapi StartGameSessionPlacementCreateGameSession, atau DescribeGameSessionpermintaan.

Tipe: Bilangan Bulat

Nilai Valid: 1900 hingga 2000.

Wajib: Ya

listenPort

Nomor port yang didengarkan client game pada pesan yang dikirim menggunakan saluran UDP.

Tipe: Bilangan Bulat

Nilai Valid: 33400 hingga 33500.

Wajib: Ya

token

Informasi opsional yang mengidentifikasi permintaan client game kepada proses server.

Tipe: ConnectionToken

Wajib: Ya

Nilai kembali

Mengembalikan nilai ConnectionStatusenum 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 kembali

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.

Tipe: Bilangan Bulat

Wajib: Ya

Nilai kembali

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 kembali

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. ID grup didefinisikan developer.

Tipe: Bilangan Bulat

Wajib: Ya

Nilai kembali

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. ID grup didefinisikan developer.

Tipe: Bilangan Bulat

Wajib: Ya

Nilai kembali

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. ID grup didefinisikan developer.

Tipe: Bilangan Bulat

Wajib: Ya

Nilai kembali

Metode ini tidak mengembalikan apa pun.

Referensi API klien Server Realtime (C #): Callback asinkron

Gunakan referensi C# Realtime Client API ini untuk membantu Anda mempersiapkan game multipemain untuk digunakan dengan Server Realtime yang digunakan di armada Amazon. GameLift Untuk informasi detail tentang proses integrasi, lihat Siapkan server Realtime.

- Tindakan Sinkron
- Callback Asinkron
- Jenis Data

Sebuah client game perlu menerapkan metode callback ini untuk menanggapi kejadian. Server Realtime memanggil callback ini untuk mengirim informasi yang berhubungan dengan game ke client game. Callback untuk kejadian yang sama juga dapat diimplementasikan dengan logika game kustom dalam skrip server Realtime. Lihat Callback skrip untuk Server Realtime.

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 kembali

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 kembali

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 kembali

Metode ini tidak mengembalikan apa pun.

OnDataReceived()

Dipanggil ketika client game menerima pesan dari server Realtime. 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 kembali

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 kembali

Metode ini tidak mengembalikan apa pun.

Referensi API klien Server Realtime (C #): Tipe data

Referensi API Klien C # Realtime ini dapat membantu Anda mempersiapkan game multipemain untuk digunakan dengan Server Realtime yang digunakan di armada Amazon. GameLift Untuk informasi detail tentang proses integrasi, lihat Siapkan server Realtime.

- Tindakan Sinkron
- Callback Asinkron
- Jenis Data

ClientConfiguration

Informasi tentang bagaimana client game menghubungkan ke server Realtime.

Daftar Isi

ConnectionType

Jenis koneksi client/server untuk digunakan, baik aman atau tidak aman. Jika Anda tidak menentukan jenis koneksi, pengaturan default-nya adalah koneksi tidak aman.

> **Note**
>
> Saat menyambung 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 game client dan/atau pemain yang meminta koneksi dengan server Realtime.

Daftar Isi

playerSessionId

ID unik yang dikeluarkan oleh Amazon GameLift saat sesi pemain baru dibuat. ID sesi pemain ditentukan dalam `PlayerSession` objek, yang dikembalikan sebagai respons terhadap panggilan klien ke tindakan GameLiftAPI StartGameSessionPlacement,, CreateGameSessionDescribeGameSessionPlacement, atau DescribePlayerSessions.

Tipe: String

Wajib: Ya

payload

Informasi yang didefinisikan developer untuk dikomunikasikan ke server Realtime mengenai koneksi. Ini mencakup data acak yang mungkin digunakan untuk mekanisme kustom untuk masuk. Sebagai contoh, muatan dapat memberikan informasi autentikasi untuk diproses oleh skrip server Realtime sebelum mengizinkan client untuk menyambung.

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 set kode operasi, tetapi dapat diubah kapan saja.

Tipe: Bilangan Bulat

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. ID grup didefinisikan developer.

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 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.

Tipe: Bilangan Bulat

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.

Tipe: Bilangan Bulat

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.

Tipe: Bilangan Bulat

Wajib: Ya

opCode

kode operasi yang didefinisikan developer yang mengidentifikasi kejadian atau tindakan game.

Tipe: Bilangan Bulat

Wajib: Ya

groupId

ID unik yang mengidentifikasi grup penerima pesan yang dikirim. ID grup didefinisikan developer.

Tipe: Bilangan Bulat

Wajib: Ya

playerId

Daftar ID pemain yang saat ini merupakan anggota grup yang ditentukan.

Jenis: himpunan integer

Wajib: Ya

enum

Enum yang didefinisikan untuk SDK Realtime Client didefinisikan sebagai berikut:

ConnectionStatus

- CONNECTED — Client game terhubung ke server Realtime dengan koneksi TCP saja. Semua pesan, terlepas dari maksud pengirimannya, dikirim melalui TCP.
- CONNECTED_SEND_FAST - Client game terhubung ke server Realtime 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 - Client game terhubung ke server Realtime dengan koneksi TCP dan UDP. Client game dapat mengirim dan menerima pesan menggunakan TCP atau UDP.
- CONNECTING client game telah mengirim permintaan koneksi dan Realtime server sedang memprosesnya.

- DISCONNECTED_CLIENT_CALL — Client game terputus dari server Realtime dalam menanggapi permintaan Disconnect() dari client game.
- DISCONNECTED — Client game terputus dari server Realtime untuk alasan selain client memutuskan panggilan.

ConnectionType

- RT_OVER_WSS_DTLS_TLS12 — Jenis koneksi aman.

  Untuk digunakan dengan server Realtime yang berjalan di GameLift 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 Amazon GameLift Realtime Server

Gunakan sumber daya ini untuk membangun logika kustom dalam skrip Realtime Anda.

Topik

- Callback skrip untuk Server Realtime
- Antarmuka Server Realtime

Callback skrip untuk Server Realtime

Anda dapat memberikan logika kustom untuk merespon kejadian dengan menerapkan callback ini dalam skrip Realtime Anda.

Init

Menginisialisasi server Realtime 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 AmazonGameLift. 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)
```

onSendToGrup

Dipanggil ketika pesan diterima pada server dari satu pemain yang akan dikirim ke grup. Proses ini berjalan sebelum pesan dikirim.

Sintaksis

```
onSendToGroup(gameMessage))
```

onPlayerJoinGrup

Dipanggil ketika seorang pemain mengirimkan permintaan untuk bergabung dengan grup.

Sintaksis

```
onPlayerJoinGroup(groupId, peerId)
```

onPlayerLeaveGrup

Dipanggil ketika seorang pemain mengirimkan permintaan untuk meninggalkan grup.

Sintaksis

```
onPlayerLeaveGroup(groupId, peerId)
```

Antarmuka Server Realtime

Ketika skrip Realtime menginisialisasi, sebuah antarmuka ke Realtime server dikembalikan. Topik ini menjelaskan properti dan metode yang tersedia melalui antarmuka. Pelajari selengkapnya tentang menulis skrip Realtime dan lihat contoh skrip detail di Membuat skrip Realtime.

Antarmuka Realtime menyediakan akses ke objek berikut:

- sesi
- pemain
- gameMessage
- konfigurasi

Objek Realtime Session

Gunakan metode ini untuk mengakses informasi terkait server dan melakukan tindakan terkait server.

getPlayers()

Mengambil daftar ID rekan untuk pemain yang saat ini terhubung ke sesi game. 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 (Siaran) dan grup untuk menerima pembaruan (targetGroupId). ID grup harus berupa bilangan bulat positif atau "-1" untuk mengindikasikan semua grup. Lihat Contoh skrip Server Realtime untuk contoh ID grup yang ditetapkan pengguna.

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 server Realtime untuk mengakhiri server game. Fungsi ini harus dipanggil dari skrip Realtime untuk keluar dengan bersih dari sesi game.

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". Misalnya: `logger.info("<string>")`.

Sintaksis

```
rtSession.getLogger()
```

sendMessage()

Mengirim pesan, yang dibuat menggunakan `newTextGameMessage` atau `newBinaryGameMessage`, dari server Realtime ke penerima pemain menggunakan saluran UDP. Mengidentifikasi penerima menggunakan ID rekan pemain.

Sintaksis

```
rtSession.sendMessage(gameMessage, targetPlayer)
```

sendGroupMessage()

Mengirim pesan, yang dibuat menggunakan `newTextGameMessage` atau `newBinaryGameMessage`, dari server Realtime ke semua pemain dalam grup pemain menggunakan saluran UDP. ID grup harus berupa bilangan bulat positif atau "-1" untuk mengindikasikan semua grup. Lihat Contoh skrip Server Realtime untuk contoh ID grup yang ditetapkan pengguna.

Sintaksis

```
rtSession.sendGroupMessage(gameMessage, targetGroup)
```

sendReliableMessage()

Mengirim pesan, yang dibuat menggunakan `newTextGameMessage` atau `newBinaryGameMessage`, dari server Realtime ke penerima pemain menggunakan koneksi TCP. Mengidentifikasi penerima menggunakan ID rekan pemain.

Sintaksis

```
rtSession.sendReliableMessage(gameMessage, targetPlayer)
```

sendReliableGroupPesan ()

Mengirim pesan, yang dibuat menggunakan `newTextGameMessage` atau `newBinaryGameMessage`, dari server Realtime ke semua pemain dalam grup pemain menggunakan saluran TCP. ID grup harus berupa bilangan bulat positif atau "-1" untuk mengindikasikan semua grup. Lihat Contoh skrip Server Realtime untuk contoh ID grup yang ditetapkan pengguna.

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 Realtime Client (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 Realtime Client (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 untuk client game saat terhubung ke server Realtime dan bergabung dengan sesi game.

pemain. playerSessionId

ID sesi pemain yang direferensikan oleh client game saat terhubung ke server Realtime dan bergabung dengan sesi game.

Objek pesan game

Gunakan metode ini untuk mengakses pesan yang diterima oleh server Realtime. 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 olehRealTimeServers.

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.

Referensi SDK GameLift server Amazon

Bagian ini berisi dokumentasi referensi untuk SDK GameLift server Amazon. Gunakan SDK server untuk mengintegrasikan server game khusus Anda untuk berkomunikasi dengan layanan AmazonGameLift.

Topik

- Referensi SDK GameLift server Amazon untuk C++
- Referensi SDK GameLift server Amazon untuk C #
- Referensi SDK GameLift server Amazon untuk Go
- Referensi SDK GameLift server Amazon untuk Unreal Engine

Referensi SDK GameLift server Amazon untuk C++

Anda dapat menggunakan referensi SDK server Amazon GameLift C++ ini untuk membantu Anda mempersiapkan game multipemain untuk digunakan dengan Amazon. GameLift Untuk detail tentang proses integrasi, lihatTambahkan Amazon GameLift ke server game Anda.

Topik

- Referensi SDK 5.x GameLift server Amazon untuk C++
- Referensi SDK GameLift 3.x server Amazon C++

Referensi SDK 5.x GameLift server Amazon untuk C++

Referensi Amazon GameLift C++ Server SDK 5.x ini dapat membantu Anda mempersiapkan game multipemain untuk digunakan dengan Amazon. GameLift Untuk detail tentang proses integrasi, lihatTambahkan Amazon GameLift ke server game Anda.

> **Note**
>
> Topik ini menjelaskan Amazon GameLift C++ API yang dapat Anda gunakan saat Anda membangun dengan C++ Standard Library ()`std`. Secara khusus, dokumentasi ini berlaku untuk kode yang Anda kompilasi dengan `-DGAMELIFT_USE_STD=1` opsi.

Topik

- GameLift Referensi SDK server Amazon (C ++) 5.x: Tindakan
- Referensi SDK GameLift server Amazon (C ++): Tipe data

GameLift Referensi SDK server Amazon (C ++) 5.x: Tindakan

Anda dapat menggunakan referensi SDK server Amazon GameLift C ++ ini untuk membantu Anda mempersiapkan game multipemain untuk digunakan dengan Amazon. GameLift Untuk detail tentang proses integrasi, lihatTambahkan Amazon GameLift ke server game Anda.

> **Note**
>
> Topik ini menjelaskan Amazon GameLift C++ API yang dapat Anda gunakan saat membuat dengan C++ Standard Library ()`std`. Secara khusus, dokumentasi ini berlaku untuk kode yang Anda kompilasi dengan `-DGAMELIFT_USE_STD=1` opsi.

Tindakan

- 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 SDK untuk armada EC2 terkelola. Panggil metode ini saat peluncuran, sebelum inisialisasi lain yang terkait dengan Amazon GameLift terjadi. Metode ini membaca parameter server dari lingkungan host untuk mengatur komunikasi antara server dan GameLift layanan Amazon.

Sintaks

```
Server::InitSDKOutcome Server::initSdkOutcome = InitSDK();
```

Nilai yang ditampilkan

Mengembalikan the section called “InitSDKOutcome” objek yang menunjukkan apakah proses server siap untuk memanggilProcessReady().

Contoh

```
//Call InitSDK to establish a local connection with the GameLift agent to enable
 further communication.
Aws::GameLift::Server::InitSDKOutcome initSdkOutcome =
  Aws::GameLift::Server::InitSDK();
```

InitSDK()

Menginisialisasi Amazon GameLift SDK untuk armadaAnywhere. Panggil metode ini saat peluncuran, sebelum inisialisasi lain yang terkait dengan Amazon GameLift terjadi. Metode ini memerlukan parameter server eksplisit untuk mengatur komunikasi antara server dan GameLift layanan Amazon.

Sintaksis

```
Server::InitSDKOutcome Server::initSdkOutcome = InitSDK(serverParameters);
```

Parameter-parameter

ServerParameters

Untuk menginisialisasi server game di GameLift Anywhere armada Amazon, 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 GameLift armada Amazon yang berisi GameLift Anywhere komputasi Amazon Anda.
- Token otorisasi yang dihasilkan oleh GameLift operasi Amazon.

Nilai yang ditampilkan

Mengembalikan the section called “InitSDKOutcome” objek yang menunjukkan apakah proses server siap untuk memanggilProcessReady().

> **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

GameLift AnywhereContoh Amazon

```
//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 GameLift agent to enable
 further communication.
Aws::GameLift::Server::InitSDKOutcome initSdkOutcome =
 Aws::GameLift::Server::InitSDK(serverParameters);
```

ProcessReady()

Memberitahu Amazon GameLift bahwa proses server siap untuk meng-host sesi game. Panggil metode ini setelah memanggilInitSDK(). Metode ini harus dipanggil hanya satu kali per proses.

Sintaksis

```
GenericOutcome ProcessReady(const Aws::GameLift::Server::ProcessParameters
&processParameters);
```

Parameter-parameter

processParameters

Sebuah objek ProcessParameters yang mengomunikasikan informasi berikut tentang proses server:

- Nama metode callback yang diimplementasikan dalam kode server game yang dipanggil GameLift layanan Amazon untuk berkomunikasi dengan proses server.
- Nomor port yang didengarkan oleh proses server.
- Jalur ke file khusus sesi game apa pun yang Anda GameLift ingin Amazon tangkap dan simpan.

Nilai yang ditampilkan

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()

Memberi tahu GameLift layanan Amazon bahwa proses server siap untuk meng-host sesi game. Metode ini harus dipanggil setelah proses server siap untuk meng-host sesi permainan. Parameter menentukan nama fungsi callback untuk Amazon GameLift untuk memanggil 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.

Sintaksis

```
GenericOutcomeCallable ProcessReadyAsync(
```

```
    const Aws::GameLift::Server::ProcessParameters &processParameters);
```

Parameter-parameter

processParameters

Sebuah objek ProcessParameters yang mengomunikasikan informasi berikut tentang proses server:

- Nama metode callback yang diimplementasikan dalam kode server game yang dipanggil GameLift layanan Amazon untuk berkomunikasi dengan proses server.
- Nomor port yang didengarkan oleh proses server.
- Jalur ke file khusus sesi game apa pun yang Anda GameLift ingin Amazon tangkap dan simpan.

Diperlukan: Ya

Nilai yang ditampilkan

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 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()

Memberi tahu Amazon GameLift 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.

Sintaksis

```
GenericOutcome
 UpdatePlayerSessionCreationPolicy(Aws::GameLift::Model::PlayerSessionCreationPolicy
 newPlayerSessionPolicy);
```

Parameter-parameter

playerCreationSessionKebijakan

Jenis: nilai `PlayerSessionCreationPolicy` enum.

Diperlukan: Ya

Nilai yang ditampilkan

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”

Sintaksis

```
AwsStringOutcome GetGameSessionId()
```

Parameter-parameter

Tindakan ini tidak memiliki parameter.

Nilai yang ditampilkan

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. Amazon GameLift `onProcessTerminate()` menyerukan alasan-alasan berikut:

- Ketika proses server telah melaporkan kesehatan yang buruk atau belum menanggapi Amazon GameLift.
- 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. `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.

Jika proses belum menerima ProcessParameters. OnProcessTerminate() callback, pesan kesalahan dikembalikan. Untuk informasi selengkapnya tentang mematikan proses server, lihatMenanggapi notifikasi shutdown proses server.

Contoh

```
Aws::GameLift::AwsLongOutcome TermTimeOutcome =
 Aws::GameLift::Server::GetTerminationTime();
```

AcceptPlayerSession()

Memberi tahu Amazon GameLift bahwa pemain dengan ID sesi pemain yang ditentukan telah terhubung ke proses server dan memerlukan validasi. Amazon GameLift memverifikasi bahwa ID sesi pemain valid. Setelah sesi pemain divalidasi, Amazon GameLift mengubah status slot pemain dari RESERVED menjadi AKTIF.

Sintaksis

```
GenericOutcome AcceptPlayerSession(String playerSessionId)
```

Parameter-parameter

playerSessionId

ID unik yang dikeluarkan oleh Amazon GameLift saat sesi pemain baru dibuat.

Nilai yang ditampilkan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Contoh ini menangani permintaan koneksi yang mencakup memvalidasi dan menolak ID sesi pemain yang tidak valid.

```
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 bahwa pemain telah terputus dari proses server. Sebagai tanggapan, Amazon GameLift mengubah slot pemain menjadi tersedia.

Sintaksis

```
GenericOutcome RemovePlayerSession(String playerSessionId)
```

Parameter-parameter

`playerSessionId`

ID unik yang dikeluarkan oleh Amazon GameLift saat sesi pemain baru dibuat.

Nilai yang ditampilkan

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

Sintaksis

```
DescribePlayerSessionsOutcome DescribePlayerSessions(DescribePlayerSessionsRequest
 describePlayerSessionsRequest)
```

Parameter-parameter

DescribePlayerSessionsRequest

the section called “DescribePlayerSessionsRequest”Objek yang menggambarkan sesi pemain mana yang akan diambil.

Nilai yang ditampilkan

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 NextTokendan menyetel nilai Batas ke 10, Amazon GameLift 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, Amazon GameLift 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.

Sintaksis

```
StartMatchBackfillOutcome StartMatchBackfill (StartMatchBackfillRequest
 startBackfillRequest);
```

Parameter-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 akan menghasilkannya.
- Matchmaker untuk dikirimi 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 ditampilkan

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.

Sintaksis

```
GenericOutcome StopMatchBackfill (StopMatchBackfillRequest stopBackfillRequest);
```

Parameter-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 yang terkait dengan permintaan isi ulang.

Nilai yang ditampilkan

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 GameLift Anywhere sumber daya komputasi Amazon Anda dan Amazon. GameLift Anda dapat menggunakan jalur sertifikat saat mendaftarkan perangkat komputasi ke GameLift Anywhere armada Amazon. Untuk informasi lebih lanjut 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 kredensyal peran IAM yang memberi wewenang kepada Amazon GameLift untuk berinteraksi dengan orang lain. Layanan AWS Untuk informasi selengkapnya, lihat Berkomunikasi dengan sumber daya AWS lain dari armada.

Sintaks

```
GetFleetRoleCredentialsOutcome GetFleetRoleCredentials(GetFleetRoleCredentialsRequest
 request);
```

Parameter-parameter

GetFleetRoleCredentialsRequest

Nilai yang ditampilkan

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 kredensyal untuk tujuan audit. Jika Anda tidak memberikan nama sesi peran, nilai default "*`[fleet-id] - [host-id]`*" akan 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 GameLift game Amazon 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, hubungi `Destroy()` lalu `InitSDK()` inisialisasi ulang sebelum memberi tahu Amazon GameLift bahwa prosesnya siap untuk meng-host sesi game. `ProcessReady()`

Sintaksis

```
GenericOutcome Aws::GameLift::Server::Destroy();
```

Parameter-parameter

Tidak ada parameter.

Nilai yang ditampilkan

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);
}
```

Referensi SDK GameLift server Amazon (C ++): Tipe data

Anda dapat menggunakan referensi SDK server Amazon GameLift C ++ ini untuk membantu Anda mempersiapkan game multipemain untuk digunakan dengan Amazon. GameLift Untuk detail tentang proses integrasi, lihatTambahkan Amazon GameLift ke server game Anda.

> **Note**
>
> Topik ini menjelaskan Amazon GameLift C++ API yang dapat Anda gunakan saat membuat dengan C++ Standard Library ()`std`. Secara khusus, dokumentasi ini berlaku untuk kode yang Anda kompilasi dengan `-DGAMELIFT_USE_STD=1` opsi.

Tipe 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

- InitSDKOutcome
- GameLiftError
- Enum

LogParameters

Objek yang mengidentifikasi file yang dihasilkan selama sesi permainan yang Anda GameLift ingin Amazon unggah dan simpan setelah sesi permainan berakhir. Server game menyediakan `LogParameters` ke Amazon GameLift sebagai bagian dari `ProcessParameters` objek dalam ProcessReady() panggilan.

Sifat-sifat	Deskripsi
LogPaths	Daftar jalur direktori ke file log server game yang Anda ingin Amazon simpan GameLift untuk akses masa depan. 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 sepertiMyGame\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 dalam fileProcessReady().

Sifat-sifat	Deskripsi

LogParameters	Objek dengan jalur direktori ke file yang dihasilkan selama sesi permainan. Amazon GameLift menyalin dan menyimpan file untuk akses future. Jenis: `Aws::GameLift::Server:: LogParameters` Wajib: Tidak
OnHealthCheck	Fungsi callback yang GameLift dipanggil Amazon untuk meminta laporan status kesehatan dari proses server. Amazon GameLift 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 mencatat proses server sebagai tidak sehat. Jenis: `std::function<bool()> onHealthCheck` Wajib: Tidak
OnProcessTerminate	Fungsi callback yang GameLift dipanggil Amazon untuk memaksa proses server dimatikan. Setelah memanggil fungsi ini, Amazon GameLift menunggu 5 menit hingga proses server dimatikan dan merespons dengan ProcessEnding() panggilan sebelum mematikan proses server. Jenis: `std::function<void()> onProcessTerminate` Wajib: Ya

OnRefreshConnection	Nama fungsi callback yang GameLift dipanggil Amazon untuk menyegarkan koneksi dengan server game. Jenis: `void OnRefreshConnectionDelegate()` Wajib: Ya
OnStartGameSession	Fungsi callback yang GameLift dipanggil Amazon untuk mengaktifkan sesi permainan baru. Amazon GameLift memanggil fungsi ini sebagai tanggapan atas permintaan klien CreateGameSession. Fungsi callback meneruskan GameSessionobjek, seperti yang didefinisikan dalam Referensi Amazon GameLift API. Jenis: `const std::function<void (Aws::GameLift::Model::GameSession)> onStartGameSession` Wajib: Ya

OnUpdateGameSession	Fungsi callback yang GameLift dipanggil Amazon untuk meneruskan objek sesi game yang diperbarui ke proses server. Amazon GameLift memanggil fungsi ini ketika permintaan pengisian ulang kecocokan telah diproses untuk menyediakan data mak comblang yang diperbarui. Ini melewati GameSessionobjek, pembaruan status (`updateReason`), dan ID tiket isi ulang pertandingan. Jenis: `std::function<void(Aws::GameLift::Server::Model::UpdateGameSession)> onUpdateGameSession` 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	GameSessionObjek yang ditentukan oleh Amazon GameLift API. `GameSession` Objek berisi properti yang menggambarkan sesi permainan. Jenis: `Aws::GameLift::Server::GameSession` Wajib: Ya
UpdateReason	Alasan bahwa sesi permainan sedang diperbarui. Jenis: `Aws::GameLift::Server::UpdateReason` Wajib: Ya
BackfillTicketId	ID tiket isi ulang yang mencoba memperbarui sesi permainan. Jenis: `std::string` Wajib: Tidak

GameSession

Tipe data ini memberikan 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>`

Properti	Deskripsi
	Jenis: `std::string` Wajib: Tidak
Nama	Label deskriptif dari sesi permainan. Jenis: `std::string` Wajib: Tidak
FleetId	Pengenal unik untuk armada tempat sesi permainan berjalan. Jenis: `std::string` Wajib: Tidak
MaximumPlayerSessionCount	Jumlah maksimum koneksi pemain ke sesi permainan. Jenis: `int` Wajib: Tidak
Port	Nomor port untuk sesi permainan. Untuk terhubung ke server GameLift game Amazon, aplikasi memerlukan alamat IP dan nomor port. Jenis: `in` Wajib: Tidak
IpAddress	Alamat IP dari sesi game. Untuk terhubung ke server GameLift game Amazon, aplikasi memerlukan alamat IP dan nomor port. Jenis: `std::string` Wajib: Tidak

Properti	Deskripsi
GameSessionData	Set properti sesi game khusus, diformat sebagai nilai string tunggal. Jenis: `std::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: `std::string` 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: `std :: vector < GameProperty >` 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` . • Armada yang tidak mendukung TLS:`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: `std::string` Wajib: Tidak

ServerParameters

Informasi yang digunakan untuk menjaga koneksi antara server game pada GameLift Anywhere armada Amazon dan GameLift layanan Amazon. Informasi ini digunakan saat meluncurkan proses server baru dengan InitSDK(). Untuk server yang dihosting di instans EC2 GameLift terkelola Amazon, gunakan objek kosong.

Properti	Deskripsi
webSocketUrl	`GameLiftServerSdkEndpoint` Amazon GameLift kembali ketika Anda `RegisterCompute` untuk sumber daya GameLift Anywhere komputasi Amazon. Jenis: `std::string`

Properti	Deskripsi
	Wajib: Ya
ProsesSid	Pengenal unik yang terdaftar pada proses server yang menghosting game Anda. Jenis: `std::string` Wajib: Ya
hostId	`HostID`Ini adalah yang `ComputeName` digunakan saat Anda mendaftarkan komputasi Anda. Untuk informasi lebih lanjut lihat, RegisterCompute. Jenis: `std::string` Wajib: Ya
FleeTID	Pengidentifikasi unik armada tempat komputasi terdaftar. Untuk informasi lebih lanjut lihat, RegisterCompute. Jenis: `std::string` Wajib: Ya
AuthToken	Token otentikasi yang dihasilkan oleh Amazon GameLift yang mengautentikasi server Anda ke Amazon. GameLift Untuk informasi lebih lanjut lihat, GetComputeAuthToken. Jenis: `std::string` Wajib: Ya

StartMatchBackfillRequest

Informasi yang digunakan untuk membuat permintaan pengisian ulang perjodohan. Server game mengkomunikasikan informasi ini ke Amazon GameLift dalam satu StartMatchBackfill() panggilan.

Properti	Deskripsi
GameSessionArn	Pengidentifikasi sesi permainan yang unik. Operasi API `GetGameSessionId` mengembalikan pengenal dalam format ARN. Jenis: `std::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: `std::string` Wajib: Ya
Pemain	Satu set data yang mewakili semua pemain yang berada di sesi permainan. Matchmake r menggunakan informasi ini untuk mencari pemain baru yang cocok untuk pemain saat ini. Jenis: `std::vector<Player>` Wajib: Ya
TicketId	Pengenal unik untuk tiket permintaan pencocokan atau pencocokan isi ulang. Jika Anda tidak memberikan nilai, Amazon GameLift menghasilkannya. Gunakan pengidentifikasi ini

Properti	Deskripsi
	untuk melacak status tiket backfill match atau membatalkan permintaan jika diperlukan. Jenis: `std::string` 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 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 mengevalu asi latensi pemain, pemain harus melaporkan latensi untuk dicocokkan. Jenis: `Dictionary<string,int>` 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. Untuk informasi selengkapnya tentang atribut pemain, lihat AttributeValue.

Properti	Deskripsi
	Jenis: `std::map<std::string,AttributeValue>` 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: `std::string` Wajib: Tidak

DescribePlayerSessionsRequest

Objek yang menentukan sesi pemain mana yang akan diambil. Proses server menyediakan informasi ini dengan DescribePlayerSessions() panggilan ke Amazon GameLift.

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`arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string>` . `GameSessionID` Ini adalah string ID kustom atau Jenis: `std::string`

Properti	Deskripsi
	Wajib: Tidak
PlayerSessionId	Pengidentifikasi unik untuk sesi pemain. Gunakan parameter ini untuk meminta satu sesi pemain tertentu. Jenis: `std::string` Wajib: Tidak
PlayerId	Pengenal unik untuk pemain. Gunakan parameter ini untuk meminta semua sesi pemain untuk pemain tertentu. Lihat Hasilkan ID pemain. Jenis: `std::string` 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: `std::string` 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: `std::string` Wajib: Tidak
Kuota	Jumlah hasil maksimum yang akan dikembali kan. Jika Anda memberikan ID sesi pemain, parameter ini diabaikan. Jenis: `int` Wajib: Tidak

StopMatchBackfillRequest

Informasi yang digunakan untuk membatalkan permintaan pengisian ulang perjodohan. Server game mengkomunikasikan informasi ini ke GameLift layanan Amazon dalam StopMatchBackfill() panggilan.

Properti	Deskripsi
GameSessionArn	Pengidentifikasi sesi permainan unik dari permintaan yang dibatalkan. Jenis: `char[]` Wajib: Tidak
MatchmakingConfigurationArn	Pengidentifikasi unik dari mak comblang permintaan ini dikirim ke. Jenis: `char[]`

Properti	Deskripsi
	Wajib: Tidak
TicketId	Pengenal unik dari tiket permintaan isi ulang yang akan dibatalkan. Jenis: `char[]` 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: • TIDAK ADA • STRING • GANDA • STRING_LIST • STRING_DOUBLE_PETA Wajib: Tidak
D	Merupakan nilai atribut string. Jenis: `std::string` Wajib: Tidak
T	Merupakan nilai atribut numerik.

Properti	Deskripsi
	Jenis: `double` Wajib: Tidak
SL	Merupakan array nilai atribut string. Jenis: `std::vector<std::string>` Wajib: Tidak
SDM	Merupakan kamus kunci string dan nilai ganda. Jenis: `std::map<std::string, double>` Wajib: Tidak

GetFleetRoleCredentialsRequest

Tipe 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.

Properti	Deskripsi
RoleArn	Nama Sumber Daya Amazon (ARN) dari peran layanan yang memperluas akses terbatas ke sumber daya Anda. AWS Jenis: `std::string` 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.

Properti	Deskripsi
	Jenis: `std::string` Wajib: Tidak

AwsLongOutcome

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 “GameLiftError” Wajib: Tidak

AwsStringOutcome

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan. Jenis: `std::string` 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 “GameLiftError” Wajib: Tidak

DescribePlayerSessionsOutcome

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan.

Properti	Deskripsi
	Jenis: the section called “DescribePlayerSessionsResult” Wajib: Tidak
ResultWithOwnership	Hasil dari tindakan, dilemparkan sebagai rvalue, sehingga kode panggilan dapat mengambil kepemilikan objek. Jenis: `Aws::GameLift::Server::Model::DescribePlayerSessionsResult&&` 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

DescribePlayerSessionsResult

Kumpulan objek yang berisi properti untuk setiap sesi pemain yang cocok dengan permintaan.

Properti	Deskripsi
NextToken	Token yang menunjukkan awal halaman hasil sekuensial berikutnya. Gunakan token yang dikembalikan dengan panggilan sebelumnya untuk operasi ini. Untuk memulai di awal set

Properti	Deskripsi
	hasil, jangan tentukan nilai. Jika ID sesi pemain ditentukan, parameter ini diabaikan. Jenis: `std::string` Wajib: Ya
PlayerSessions	Jenis: `IList<`the section called “PlayerSession” `>` Diperlukan:
ResultWithOwnership	Hasil dari tindakan, dilemparkan sebagai rvalue, sehingga kode panggilan dapat mengambil kepemilikan objek. Jenis: `std::string&&` 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

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 GameLift diteruskan Amazon ke server game. Untuk informasi lebih lanjut, 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 PlayerSessionStatu sFilter berpengaruh pada respons. Jenis: A `PlayerSessionStatus` enum. Kemungkinan nilainya mencakup berikut ini:

Properti	Deskripsi
	• AKTIF • 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

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

StartMatchBackfillResult

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
TicketId	Pengenal unik untuk tiket perjodohan. Jika tidak ada ID tiket yang ditentukan di sini, Amazon GameLift akan menghasilkan satu dalam bentuk UUID. Gunakan pengenal ini untuk melacak status tiket isi ulang pertandingan dan mengambil hasil pertandingan. Jenis: `std::string` Wajib: Tidak

GetComputeCertificateOutcome

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan.

Properti	Deskripsi
	Jenis: the section called “GetComputeCertificateResult” Wajib: Tidak
ResultWithOwnership	Hasil dari tindakan, dilemparkan sebagai rvalue, sehingga kode panggilan dapat mengambil kepemilikan objek. Jenis: `Aws::GameLift::Server::Model::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	Jalur ke sertifikat TLS pada sumber daya komputasi Anda. Saat menggunakan armada GameLift terkelola Amazon, jalur ini berisi:

Properti	Deskripsi
	• `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. Jenis: `std::string` Wajib: Tidak
ComputeName	Nama sumber daya komputasi Anda. Jenis: `std::string` 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

Properti	Deskripsi
ResultWithOwnership	Hasil dari tindakan, dilemparkan sebagai rvalue, sehingga kode panggilan dapat mengambil kepemilikan objek. Jenis: `Aws::GameLift::Server::Model::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`

Properti	Deskripsi
	Wajib: Tidak
AssumedRoleUserArn	Nama Sumber Daya Amazon (ARN) pengguna yang menjadi milik peran layanan. Jenis: `string` Wajib: Tidak
Kedaluwarsa	Jumlah waktu hingga kredensi sesi Anda kedaluwarsa. Jenis: `DateTime` Wajib: Tidak
SecretAccessKey	ID kunci akses rahasia untuk otentikasi. Jenis: `string` Wajib: Tidak
SessionToken	Token untuk mengidentifikasi sesi aktif saat ini yang berinteraksi dengan AWS sumber daya Anda. Jenis: `string` Wajib: Tidak
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: `bool` Wajib: Ya

Properti	Deskripsi
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Jenis: the section called “GameLiftError” Wajib: Tidak

InitSDKOutcome

> **Note**
>
> `InitSDKOutcome`dikembalikan 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: `bool` Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Jenis: the section called “GameLiftError” Wajib: Tidak

GameLiftError

Properti	Deskripsi
ErrorType	Jenis kesalahan.

Properti	Deskripsi
	Jenis: A `GameLiftErrorType` enum. Wajib: Tidak
ErrorName	Nama kesalahannya. Jenis: `std::string` Wajib: Tidak
ErrorMessage	Pesan kesalahan. Jenis: `std::string` Wajib: Tidak

Enum

Enum yang didefinisikan untuk SDK GameLift server Amazon (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 ke Amazon gagal. GameLift
- NETWORK_NOT_INITIALIZED — Jaringan belum diinisialisasi.
- SERVICE_CALL_FAILED - Panggilan ke layanan telah gagal. AWS
- WEBSOCKET_CONNECT_FAILURE
- WEBSOCKET_CONNECT_FALURE_FORBIDDEN
- WEBSOCKET_CONNECT_FAILURE_INVALID_URL
- WEBSOCKET_CONNECT_FAILURE_TIMEOUT
- ALREADY_INITIALIZED — Server GameLift Amazon atau Klien telah diinisialisasi dengan Initialize ().

- FLEET_MISMATCH — Armada target tidak cocok dengan armada GameSession atau PlayerSession.
- GAMELIFT_CLIENT_NOT_INITIALIZED - Klien Amazon belum diinisialisasi. GameLift
- GAMELIFT_SERVER_NOT_INITIALIZED — Server Amazon belum diinisialisasi. GameLift
- GAME_SESSION_ENDED_FAILED — GameLift Amazon Server SDK tidak dapat menghubungi layanan untuk melaporkan sesi permainan berakhir.
- GAME_SESSION_NOT_READY — Sesi Game Server GameLift Amazon tidak diaktifkan.
- GAME_SESSION_READY_FAILED — GameLift Amazon Server SDK tidak dapat menghubungi layanan untuk melaporkan sesi permainan sudah siap.
- INITIALIZATION_MISMATCH — Metode klien dipanggil setelah Server: :Initialize (), atau sebaliknya.
- NOT_INITIALIZED — GameLift Server Amazon 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 - GameLift Amazon Server SDK tidak 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 - GameLift Amazon Server SDK tidak dapat menghubungi layanan untuk melaporkan proses siap.
- SDK_VERSION_DETECTION_FAILED — Deteksi versi SDK gagal.
- STX_CALL_FAILED — Panggilan ke komponen backend server XSTx telah gagal.
- STX_INITIALIZATION_FAILED - Komponen backend server XSTx gagal menginisialisasi.
- UNEXPECTED_PLAYER_SESSION — Sesi pemain yang tidak terdaftar ditemui oleh server.
- WEBSOCKET_CONNECT_FAILURE
- WEBSOCKET_CONNECT_FALURE_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.

Referensi SDK GameLift 3.x server Amazon C++

Anda dapat menggunakan referensi SDK 3.x server Amazon GameLift C++ ini untuk membantu Anda mempersiapkan game multipemain untuk digunakan dengan Amazon. GameLift Untuk detail tentang proses integrasi, lihatTambahkan Amazon GameLift ke server game Anda.

Topik

- Referensi SDK (C ++) GameLift server Amazon: Tindakan
- Referensi SDK GameLift server Amazon (C ++): Jenis data

Referensi SDK (C ++) GameLift server Amazon: Tindakan

Anda dapat menggunakan referensi SDK server Amazon GameLift C++ ini untuk membantu Anda mempersiapkan game multipemain untuk digunakan dengan Amazon. GameLift Untuk detail tentang proses integrasi, lihatTambahkan Amazon GameLift ke server game Anda.

Tindakan

- AcceptPlayerSession()
- ActivateGameSession()
- DescribePlayerSessions()

- GetGameSessionId()
- GetInstanceCertificate()
- GetSdkVersion()
- GetTerminationTime()
- InitSDK()
- ProcessEnding()
- ProcessReady()
- ProcessReadyAsync()
- RemovePlayerSession()
- StartMatchBackfill()
- StopMatchBackfill()
- TerminateGameSession()
- UpdatePlayerSessionCreationPolicy()
- menghancurkan ()

AcceptPlayerSession()

Memberitahu GameLift layanan Amazon bahwa pemain dengan ID sesi pemain tertentu telah terhubung ke proses server dan memerlukan validasi. Amazon GameLift memverifikasi bahwa ID sesi pemain valid—yaitu, ID pemain telah memesan slot pemain dalam sesi permainan. Setelah divalidasi, Amazon GameLift mengubah status slot pemain dari RESERVED menjadi AKTIF.

Sintaks

```
GenericOutcome AcceptPlayerSession(const std::string& playerSessionId);
```

Parameter

playerSessionId

ID unik yang dikeluarkan oleh GameLift layanan Amazon sebagai respons terhadap panggilan ke tindakan AWS CreatePlayerSessionSDK Amazon GameLift API. Client game mereferensi ID ini saat menghubungkan ke proses server.

Jenis: std::string

Wajib: Ya

Nilai kembali

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 ID sesi pemain yang tidak valid.

```
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()

Memberitahu GameLift layanan Amazon bahwa proses server telah memulai sesi permainan dan sekarang siap untuk 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 kembali

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 kembali

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 PlayerSessiondata AWS SDK Amazon GameLift API.

Contoh

Contoh ini menggambarkan permintaan untuk semua sesi pemain yang secara aktif terhubung ke sesi game tertentu. Dengan menghilangkan `NextToken` dan menetapkan `Limit` nilai ke 10, Amazon GameLift mengembalikan 10 rekaman sesi 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 kembali `Success` = `True` dan `GameSessionId` = `""` (string kosong).

Sintaks

```
AwsStringOutcome GetGameSessionId();
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai kembali

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-yang dikaitkan dengan armada dan instansinya. AWS Certificate Managermenghasilkan sertifikat ini ketika Anda membuat armada baru dengan konfigurasi sertifikat diatur 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 kembali

Jika berhasil, mengembalikan `GetInstanceCertificateOutcome` objek yang berisi lokasi file sertifikat TLS armada dan rantai sertifikat, yang disimpan pada instance. File sertifikat root, yang diekstrak dari rantai sertifikat, juga disimpan pada instance. Jika tidak berhasil, ini mengembalikan pesan kesalahan.

Untuk informasi selengkapnya tentang data sertifikat dan 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 kembali

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()` callback dari GameLift layanan Amazon. Amazon GameLift dapat memanggil `onProcessTerminate()` karena alasan berikut: (1) ketika proses server melaporkan kesehatan yang buruk atau tidak merespons AmazonGameLift, (2) saat mengakhiri instans selama peristiwa penurunan skala, atau (3) saat instans 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 kembali

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 SDK. Metode ini harus dipanggil pada peluncuran, sebelum inisialisasi GameLift terkait Amazon lainnya terjadi.

Sintaks

```
InitSDKOutcome InitSDK();
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai kembali

Jika berhasil, mengembalikan InitSdkOutcome objek yang menunjukkan bahwa proses server siap untuk memanggilProcessReady().

Contoh

```
Aws::GameLift::Server::InitSDKOutcome initOutcome =
    Aws::GameLift::Server::InitSDK();
```

ProcessEnding()

Memberitahu GameLift layanan Amazon 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 kembali

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 GameLift layanan Amazon bahwa proses server siap untuk menyelenggarakan sesi game. 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.

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 GameLift layanan Amazon untuk berkomunikasi dengan proses server.
- Nomor port yang didengarkan oleh proses server.
- Jalur ke file khusus sesi game apa pun yang Anda GameLift ingin Amazon tangkap dan simpan.

Wajib: Ya

Nilai kembali

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 GameLift layanan Amazon bahwa proses server siap untuk menyelenggarakan sesi game. Metode ini harus dipanggil setelah proses server siap untuk menjadi host sesi game. Parameter menentukan nama fungsi callback untuk Amazon GameLift 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 GameLift layanan Amazon untuk berkomunikasi dengan proses server.
- Nomor port yang didengarkan oleh proses server.
- Jalur ke file khusus sesi game apa pun yang Anda GameLift ingin Amazon tangkap dan simpan.

Wajib: Ya

Nilai kembali

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()

Memberitahu GameLift layanan Amazon bahwa pemain dengan ID sesi pemain tertentu telah terputus dari proses server. Sebagai tanggapan, Amazon GameLift mengubah slot pemain menjadi tersedia, yang memungkinkannya ditugaskan ke pemain baru.

Sintaks

```
GenericOutcome RemovePlayerSession(
    const std::string& playerSessionId);
```

Parameter

playerSessionId

ID unik yang dikeluarkan oleh GameLift layanan Amazon sebagai respons terhadap panggilan ke tindakan AWS CreatePlayerSessionSDK Amazon GameLift API. Client game mereferensi ID ini saat menghubungkan ke proses server.

Jenis: std::string

Wajib: Ya

Nilai kembali

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 denganFlexMatch. Lihat juga tindakan 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, GameLift layanan Amazon mengirimkan data mak comblang yang diperbarui dengan menjalankan 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

```
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 bersifat opsional; jika tidak ada ID yang disediakan, Amazon GameLift akan membuat satu secara otomatis.
- Matchmaker untuk dikirimi 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 kembali

Mengembalikan StartMatchBackfillOutcome objek dengan tiket pertandingan pengurukan 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 tindakan 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 dikirimi permintaan backfill
- sesi game yang terkait dengan permintaan backfill

Wajib: Ya

Nilai kembali

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. Sebagai gantinya, proses server harus menelepon ProcessEnding() setelah sesi permainan berakhir.

Memberitahu GameLift layanan Amazon bahwa proses server telah mengakhiri sesi game 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

sinyal ke Amazon GameLift bahwa proses server segera tersedia untuk menjadi tuan rumah 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 kembali

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 tindakan 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 kembali

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
```

menghancurkan ()

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 kembali

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);
}
```

Referensi SDK GameLift server Amazon (C ++): Jenis data

Anda dapat menggunakan referensi SDK server Amazon GameLift C++ ini untuk membantu Anda mempersiapkan game multipemain untuk digunakan dengan Amazon. GameLift Untuk detail tentang proses integrasi, lihatTambahkan Amazon GameLift ke server game Anda.

API ini didefinisikan dalam `GameLiftServerAPI.h`, `LogParameters.h`, dan `ProcessParameters.h`.

- Tindakan
- Jenis Data

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.
- Menyediakan PlayerId untuk meminta semua sesi pemain untuk pemain tertentu.

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 NextTokenuntuk 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. ID Pemain ditentukan oleh developer. Lihat Hasilkan ID pemain.

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 untuk mengunggah dan menyimpan setelah sesi permainan berakhir. Informasi ini dikomunikasikan ke GameLift layanan Amazon dalam ProcessReady() panggilan.

Daftar Isi

logPaths

Jalur direktori ke file log server game yang ingin disimpan Amazon GameLift untuk akses di masa mendatang. 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 mutlak. 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 GameLift layanan Amazon 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.

Tipe: Bilangan Bulat

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 dipanggil GameLift layanan Amazon untuk mengaktifkan sesi game baru. Amazon GameLift memanggil fungsi ini sebagai respons terhadap permintaan klien CreateGameSession. Fungsi callback meneruskan GameSessionobjek (didefinisikan dalam Amazon GameLift Service API Reference).

Tipe: `const std::function<void(Aws::GameLift::Model::GameSession)> onStartGameSession`

Wajib: Ya

onProcessTerminate

Nama fungsi callback yang dipanggil GameLift layanan Amazon untuk memaksa proses server dimatikan. Setelah memanggil fungsi ini, Amazon GameLift 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 dipanggil GameLift layanan Amazon untuk meminta laporan status kesehatan dari proses server. Amazon GameLift memanggil fungsi ini setiap 60 detik. Setelah memanggil fungsi ini Amazon GameLift 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 dipanggil GameLift layanan Amazon untuk meneruskan objek sesi game yang diperbarui ke proses server. Amazon GameLift memanggil fungsi ini ketika permintaan isi ulang pertandingan telah diproses untuk menyediakan data mak comblang yang diperbarui. Ini melewati GameSessionobjek, pembaruan status (`updateReason`), dan ID tiket isi 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 ke GameLift layanan Amazon dalam StartMatchBackfill() panggilan.

Daftar Isi

GameSessionArn

Pengidentifikasi sesi game yang unik. Tindakan API GetGameSessionId() mengembalikan pengidentifikasi dalam format ARN.

Tipe: String

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.

Tipe: String

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 Panduan Referensi GameLift API Amazon untuk deskripsi format objek Player. Untuk menemukan atribut pemain, ID, dan tugas tim, lihat di objek sesi game, di properti data matchmaker. 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 disediakan di sini, Amazon GameLift 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 ke GameLift layanan Amazon dalam StopMatchBackfill() panggilan.

Daftar Isi

GameSessionArn

Pengidentifikasi sesi game unik yang terkait dengan permintaan yang dibatalkan.

Tipe: String

Wajib: Ya

MatchmakingConfigurationArn

Pengidentifikasi unik dari matchmaker sebagai tujuan pengiriman permintaan ini.

Tipe: String

Wajib: Ya

TicketId

Pengidentifikasi unik dari tiket backfill match yang akan dibatalkan.

Tipe: String

Wajib: Ya

Referensi SDK GameLift server Amazon untuk C

Anda dapat menggunakan referensi SDK server Amazon GameLift C# ini untuk membantu Anda mempersiapkan game multipemain untuk digunakan dengan Amazon. GameLift Untuk detail tentang proses integrasi, lihatTambahkan Amazon GameLift ke server game Anda.

Topik

- Referensi SDK 5.x GameLift server Amazon untuk C # dan Unity
- Referensi SDK 4.x GameLift server Amazon untuk C #

Referensi SDK 5.x GameLift server Amazon untuk C # dan Unity

Anda dapat menggunakan referensi SDK 5.x server Amazon GameLift C# ini untuk membantu Anda mempersiapkan game multipemain untuk digunakan dengan Amazon. GameLift Untuk detail tentang proses integrasi, lihat Tambahkan Amazon GameLift ke server game Anda dan untuk informasi tentang penggunaan plugin SDK server C# untuk Unity, lihat. Integrasikan Amazon GameLift ke dalam proyek Unity GameLiftServer Amazon SDK 5.x untuk C # mendukung .NET 4.6 dan .NET 6.

Topik

- Referensi SDK GameLift server Amazon untuk C# dan Unity: Tindakan
- Referensi SDK GameLift server Amazon untuk C# dan Unity: Tipe data

Referensi SDK GameLift server Amazon untuk C# dan Unity: Tindakan

Referensi SDK server Amazon GameLift C # ini membantu Anda mempersiapkan game multipemain untuk digunakan dengan Amazon. GameLift Untuk detail tentang proses integrasi, lihat Tambahkan

Amazon GameLift ke server game Anda dan untuk informasi tentang penggunaan plugin SDK server C# untuk Unity, lihat. Integrasikan Amazon GameLift ke dalam proyek Unity

Tindakan

- 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 SDK untuk armada EC2 terkelola. Panggil metode ini saat peluncuran, sebelum inisialisasi lain yang terkait dengan Amazon GameLift terjadi. Metode ini membaca parameter server dari lingkungan host untuk mengatur komunikasi antara server dan GameLift layanan Amazon.

Sintaks

```
GenericOutcome InitSDK();
```

Nilai yang ditampilkan

Jika berhasil, mengembalikan InitSdkOutcome objek untuk menunjukkan bahwa proses server siap untuk memanggilProcessReady().

Contoh

```
//Call InitSDK to establish a local connection with the GameLift agent to enable
 further communication.
GenericOutcome initSDKOutcome = GameLiftServerAPI.InitSDK();
```

InitSDK()

Menginisialisasi Amazon GameLift SDK untuk armadaAnywhere. Panggil metode ini saat peluncuran, sebelum inisialisasi lain yang terkait dengan Amazon GameLift terjadi. Metode ini memerlukan parameter server eksplisit untuk mengatur komunikasi antara server dan GameLift layanan Amazon.

Sintaksis

```
GenericOutcome InitSDK(ServerParameters serverParameters);
```

Parameter-parameter

ServerParameters

Untuk menginisialisasi server game di GameLift Anywhere armada Amazon, 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 GameLift armada Amazon yang berisi GameLift Anywhere komputasi Amazon Anda.
- Token otorisasi yang dihasilkan oleh GameLift operasi Amazon.

Nilai yang ditampilkan

Jika berhasil, mengembalikan InitSdkOutcome objek untuk menunjukkan bahwa proses server siap untuk memanggilProcessReady().

> **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 bahwa proses server siap untuk meng-host sesi game. Panggil metode ini setelah memanggilInitSDK(). Metode ini harus dipanggil hanya satu kali per proses.

Sintaksis

```
GenericOutcome ProcessReady(ProcessParameters processParameters)
```

Parameter-parameter

ProcessParameters

Sebuah `ProcessParameters` objek menyimpan informasi tentang proses server.

Nilai yang ditampilkan

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()

Memberi tahu Amazon GameLift 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()

Memberi tahu Amazon GameLift 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.

Sintaksis

```
GenericOutcome UpdatePlayerSessionCreationPolicy(PlayerSessionCreationPolicy
 playerSessionPolicy)
```

Parameter-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 ditampilkan

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. Amazon GameLift on`ProcessTerminate()` menyerukan alasan-alasan berikut:

- Ketika proses server telah melaporkan kesehatan yang buruk atau belum menanggapi Amazon GameLift.
- 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 `637355968000000000` kutu. Jika tidak ada waktu penghentian tersedia, pesan kesalahan ditampilkan.

Contoh

```
AwsDateTimeOutcome getTerminationTimeOutcome = GameLiftServerAPI.GetTerminationTime();
```

AcceptPlayerSession()

Memberi tahu Amazon GameLift bahwa pemain dengan ID sesi pemain yang ditentukan telah terhubung ke proses server dan memerlukan validasi. Amazon GameLift memverifikasi bahwa ID sesi pemain valid. Setelah sesi pemain divalidasi, Amazon GameLift mengubah status slot pemain dari RESERVED menjadi AKTIF.

Sintaksis

```
GenericOutcome AcceptPlayerSession(String playerSessionId)
```

Parameter-parameter

playerSessionId

ID unik yang dikeluarkan oleh GameLift saat sesi pemain baru dibuat.

Nilai yang ditampilkan

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 ID sesi pemain yang tidak valid.

```
void ReceiveConnectingPlayerSessionID (Connection connection, String playerSessionId)
{
  GenericOutcome acceptPlayerSessionOutcome =
 GameLiftServerAPI.AcceptPlayerSession(playerSessionId);
  if(acceptPlayerSessionOutcome.Success)
  {
    connectionToSessionMap.emplace(connection, playerSessionId);
    connection.Accept();
  }
  else
  {
    connection.Reject(acceptPlayerSessionOutcome.Error.ErrorMessage);
  }
}
```

RemovePlayerSession()

Memberitahu Amazon GameLift bahwa pemain telah terputus dari proses server. Sebagai tanggapan, Amazon GameLift mengubah slot pemain menjadi tersedia.

Sintaksis

```
GenericOutcome RemovePlayerSession(String playerSessionId)
```

Parameter-parameter

playerSessionId

ID unik yang dikeluarkan oleh Amazon GameLift saat sesi pemain baru dibuat.

Nilai yang ditampilkan

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.

Sintaksis

```
DescribePlayerSessionsOutcome DescribePlayerSessions(DescribePlayerSessionsRequest
 describePlayerSessionsRequest)
```

Parameter-parameter

DescribePlayerSessionsRequest

the section called “DescribePlayerSessionsRequest”Objek yang menggambarkan sesi pemain mana yang akan diambil.

Nilai yang ditampilkan

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 NextTokendan menyetel nilai Batas ke 10, Amazon GameLift 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, Amazon GameLift 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.

Sintaksis

```
StartMatchBackfillOutcome StartMatchBackfill (StartMatchBackfillRequest
 startBackfillRequest);
```

Parameter-parameter

StartMatchBackfillRequest

Sebuah `StartMatchBackfillRequest` objek menyimpan informasi tentang permintaan isi ulang.

Nilai yang ditampilkan

Mengembalikan the section called “StartMatchBackfillOutcome” objek dengan ID tiket isi ulang kecocokan, 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.

Sintaksis

```
GenericOutcome StopMatchBackfill (StopMatchBackfillRequest stopBackfillRequest);
```

Parameter-parameter

StopMatchBackfillRequest

`StopMatchBackfillRequest`Objek yang memberikan detail tentang tiket perjodohan yang Anda hentikan.

Nilai yang ditampilkan

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 GameLift Anywhere armada Amazon. Untuk informasi lebih lanjut 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 GameLift terkelola Amazon, 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 kredensyal peran IAM yang memberi wewenang kepada Amazon GameLift untuk berinteraksi dengan orang lain. Layanan AWS Untuk informasi selengkapnya, lihat Berkomunikasi dengan sumber daya AWS lain dari armada.

Sintaks

```
GetFleetRoleCredentialsOutcome GetFleetRoleCredentials(GetFleetRoleCredentialsRequest
 request);
```

Parameter-parameter

GetFleetRoleCredentialsRequest

Kredensyal peran yang memperluas akses terbatas ke AWS sumber daya Anda ke server game.

Nilai yang ditampilkan

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 GameLift game Amazon 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, hubungi `Destroy()` dan kemudian `InitSDK()` inisialisasi ulang sebelum memberi tahu Amazon GameLift bahwa prosesnya siap untuk meng-host 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);
```

```
}
```

Referensi SDK GameLift server Amazon untuk C# dan Unity: Tipe data

Referensi Amazon GameLift C# Server SDK ini dapat membantu Anda mempersiapkan game multipemain untuk digunakan dengan Amazon. GameLift Untuk detail tentang proses integrasi, lihat Tambahkan Amazon GameLift ke server game Anda dan untuk informasi tentang penggunaan plugin SDK server C# untuk Unity, lihat. Integrasikan Amazon GameLift ke dalam proyek Unity

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 agar server game diunggah ke Amazon GameLift setelah sesi permainan berakhir. Server game mengkomunikasikan `LogParameters to` Amazon GameLift dalam ProcessReady() panggilan.

Properti	Deskripsi
LogPaths	Daftar jalur direktori ke file log server game yang ingin disimpan Amazon GameLift untuk akses di masa mendatang. Proses server menghasilkan file-file ini selama setiap sesi permainan. Anda menentukan path file dan nama di server game Anda dan menyimpannya di direktori build game root. Jalur log harus mutlak. Misalnya, jika build game Anda menyimpan log sesi game di jalur sepertiMyGame\sessionLogs\ , maka jalurnya akan berada `c:\game\MyGame\sessionLogs` di instance Windows. Tipe: `List<String>` Wajib: Tidak

ProcessParameters

Tipe data ini berisi kumpulan parameter yang dikirim ke Amazon GameLift dalam ProcessReady() panggilan.

Properti	Deskripsi

LogParameters	Objek dengan daftar jalur direktori ke file log sesi game. Jenis: `Aws::GameLift::Server::` LogParameters Wajib: Ya
OnHealthCheck	Nama fungsi callback yang GameLift dipanggil Amazon untuk meminta laporan status kesehatan dari proses server. Amazon GameLift memanggil fungsi ini setiap 60 detik. Setelah memanggil fungsi ini Amazon GameLift menunggu 60 detik untuk respons, jika tidak ada yang diterima, Amazon GameLift mencatat proses server sebagai tidak sehat. Jenis: `void OnHealthCheckDelegate()` Wajib: Ya
OnProcessTerminate	Nama fungsi callback yang GameLift dipanggil Amazon untuk memaksa proses server dimatikan. Setelah memanggil fungsi ini, Amazon GameLift menunggu lima menit hingga proses server dimatikan dan merespons dengan ProcessEnding() panggilan sebelum mematikan proses server. Jenis: `void OnProcessTerminateDelegate()` Wajib: Ya

OnStartGameSession	Nama fungsi callback yang GameLift dipanggil Amazon untuk mengaktifkan sesi game baru. Amazon GameLift memanggil fungsi ini sebagai respons terhadap permintaan klien CreateGameSession. Fungsi callback mengambil GameSessionobjek yang ditentukan dalam Amazon GameLift API Reference. Jenis: `void OnStartGameSession Delegate( GameSession )` Wajib: Ya
OnUpdateGameSession	Nama fungsi callback yang GameLift dipanggil Amazon untuk meneruskan objek sesi game yang diperbarui ke proses server. Amazon GameLift memanggil fungsi ini ketika permintaa n isi ulang pertandingan telah diproses untuk menyediakan data mak comblang yang diperbarui. Ini melewati GameSessionobjek, 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 dikonfigu rasi 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 terbaru untuk objek sesi permainan, termasuk alasan bahwa sesi permainan diperbarui. Jika pembaruan terkait dengan tindakan isi ulang pertandingan, tipe data ini menyertakan ID tiket isi ulang.

Properti	Deskripsi
GameSession	GameSessionObjek yang ditentukan oleh Amazon GameLift API. `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 mencoba memperbarui sesi permainan.

Properti	Deskripsi
	Jenis: `String` Wajib: Ya

GameSession

Rincian sesi permainan.

Properti	Deskripsi
GameSessionId	Sebuah identifier unik untuk sesi permainan. Sebuah sesi permainan ARN memiliki format berikut:`arn:aws:gamelift:<region>::gamesession/<fleet ID>/<custom ID string or idempotency token>` . Tipe: `String` Wajib: Tidak
Nama	Label deskriptif sesi permainan. Tipe: `String` Wajib: Tidak
FleetId	Sebuah identifier unik untuk armada bahwa sesi permainan berjalan pada. Tipe: `String` Wajib: Tidak
MaximumPlayerSessionCount	Jumlah maksimum koneksi pemain ke sesi permainan. Tipe: `Integer`

Properti	Deskripsi
	Wajib: Tidak
Port	Nomor port untuk sesi permainan. Untuk terhubung ke server GameLift game Amazon, aplikasi memerlukan alamat IP dan nomor port. Tipe: `Integer` Wajib: Tidak
IpAddress	Alamat IP dari sesi game. Untuk terhubung ke server GameLift game Amazon, aplikasi memerlukan alamat IP dan nomor port. Tipe: `String` Wajib: Tidak
GameSessionData	Set properti sesi game khusus, diformat sebagai nilai string tunggal. Tipe: `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. Tipe: `String` Wajib: Tidak

Properti	Deskripsi
GameProperties	Satu set properti kustom untuk sesi permainan, diformat sebagai pasangan key:value. Properti ini dilewatkan dengan permintaan untuk memulai sesi permainan baru. Tipe: `Dictionary<string, string>` Wajib: Tidak
DnsName	DNS identifier ditugaskan untuk instance yang menjalankan sesi permainan. Nilai memiliki format berikut: • Armada yang mendukung TLS:`<unique identifier>.<region identifier>.amazongamelift.com` . • Armada yang tidak mendukung TLS:`ec2-<unique identifier>.compute.amazonaws.com` . Saat menghubungkan ke sesi game yang berjalan di armada berkemampuan TLS, Anda harus menggunakan nama DNS, bukan alamat IP. Tipe: `String` Wajib: Tidak

ServerParameters

Informasi yang digunakan untuk menjaga koneksi antara GameLift Anywhere server Amazon dan GameLift layanan Amazon. Informasi ini digunakan saat meluncurkan proses server baru dengan InitSDK(). Untuk server yang dihosting di instans EC2 GameLift terkelola Amazon, gunakan objek kosong.

Properti	Deskripsi
WebSocketUrl	`GameLiftServerSdkEndpoint` Kembali ketika Anda `RegisterCompute` sebagai bagian dari Amazon GameLiftAnywhere. Jenis: `String` Wajib: Ya
ProcessId	Pengenal unik yang terdaftar pada proses server hosting game Anda. Jenis: `String` Wajib: Ya
HostId	Pengenal unik untuk host dengan proses server hosting game Anda. HostID adalah yang ComputeName digunakan saat Anda mendaftar kan komputasi Anda. Untuk informasi lebih lanjut lihat, RegisterCompute Jenis: `String` Wajib: Ya
FleetId	ID armada armada tempat komputasi terdaftar . Untuk informasi lebih lanjut lihat, RegisterC ompute. Jenis: `String` Wajib: Ya
AuthToken	Token autentikasi yang dihasilkan oleh Amazon GameLift yang mengautentikasi server Anda ke AmazonGameLift. Untuk informasi lebih lanjut lihat, GetComputeAuthToken.

Properti	Deskripsi
	Jenis: `String` Wajib: Ya

StartMatchBackfillRequest

Informasi yang digunakan untuk membuat permintaan pengurukan perjodohan. Server game mengkomunikasikan informasi ini ke Amazon GameLift dalam StartMatchBackfill() panggilan.

Properti	Deskripsi
GameSessionArn	Pengenal sesi permainan yang unik. Operasi API `GetGameSessionId` mengembalikan identifier dalam format ARN. Jenis: `String` Wajib: Ya
MatchmakingConfigurationArn	Pengenal unik, dalam bentuk ARN, untuk mak comblang untuk digunakan untuk permintaa n ini. ARN mak comblang untuk sesi game asli ada di objek sesi game 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.

Properti	Deskripsi
	Jenis: `List<Player>` Wajib: Ya
TicketId	Pengenal unik untuk tiket permintaan perjodohan atau pencocokan isi ulang. Jika Anda tidak memberikan nilai, Amazon GameLift menghasilkan satu. Gunakan pengidentifikasi ini untuk melacak status tiket backfill match atau membatalkan permintaan jika diperlukan. Tipe: `String` Wajib: Tidak

Pemain

Merupakan pemain dalam perjodohan. Saat permintaan perjodohan dimulai, pemain memiliki ID pemain, atribut, dan kemungkinan data latensi. Amazon GameLift menambahkan informasi tim setelah pertandingan dibuat.

Properti	Deskripsi
LatencyInNONA	Satu set nilai yang dinyatakan dalam milidetik, yang menunjukkan jumlah latensi yang dialami pemain saat terhubung ke lokasi. Jika properti ini digunakan, pemain hanya cocok untuk lokasi yang terdaftar. Jika makcomblang memiliki aturan yang mengevaluasi latensi pemain, pemain harus melaporkan latensi untuk dicocokkan. Tipe: `Dictionary<string, int>` Wajib: Tidak

Properti	Deskripsi
PlayerAttributes	Kumpulan pasangan key:value yang berisi informasi pemain untuk digunakan dalam perjodohan. Kunci atribut pemain harus cocok dengan yang PlayerAttributes digunakan dalam kumpulan aturan perjodohan. Untuk informasi selengkapnya tentang atribut pemain, lihat AttributeValue. Tipe: `Dictionary<string, AttributeValue` Wajib: Tidak
PlayerId	Sebuah identifier unik untuk pemain. Tipe: `String` Wajib: Tidak
Tim	Nama tim yang ditugaskan pemain dalam pertandingan. Anda menentukan nama tim dalam set aturan perjodohan. Tipe: `String` Wajib: Tidak

DescribePlayerSessionsRequest

Jenis data ini digunakan untuk menentukan sesi pemain untuk diambil. Hal 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 tertentu. Untuk koleksi sesi pemain yang besar, gunakan parameter pemberian nomor halaman untuk mengambil hasil dalam halaman berurutan.

Properti	Deskripsi
GameSessionId	Pengenal sesi permainan 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
PlayerSessionId	Pengenal unik untuk sesi pemain. Tipe: `String` Wajib: Tidak
PlayerId	Pengenal unik untuk pemain. Lihat Hasilkan ID pemain. Tipe: `String` Wajib: Tidak
PlayerSessionStatusFilter	Status sesi pemain untuk menyaring 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.

Properti	Deskripsi
	• 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
NextToken	Token menunjukkan awal halaman berikutnya hasil. Untuk menentukan awal set hasil, jangan berikan nilai. Jika Anda memberikan ID sesi pemain, parameter ini akan diabaikan. Tipe: `String` Wajib: Tidak
Kuota	Jumlah hasil maksimum yang akan dikembali kan. Jika Anda memberikan ID sesi pemain, parameter ini akan diabaikan. Tipe: `int` Wajib: Tidak

StopMatchBackfillRequest

Informasi yang digunakan untuk membatalkan permintaan pengurukan perjodohan. Server game mengkomunikasikan informasi ini ke GameLift layanan Amazon dalam StopMatchBackfill() panggilan.

Properti	Deskripsi
GameSessionArn	Pengenal sesi permainan unik dari permintaan yang dibatalkan. Jenis: `string` Wajib: Ya
MatchmakingConfigurationArn	Pengenal unik dari mak comblang permintaan ini dikirim ke. Jenis: `string` Wajib: Ya
TicketId	Pengenal unik dari tiket permintaan isi ulang yang akan dibatalkan. Jenis: `string` Wajib: Ya

GetFleetRoleCredentialsRequest

Tipe data ini memberikan server game akses terbatas ke AWS sumber daya Anda yang lain. Untuk informasi lebih lanjut lihat,Menyiapkan peran layanan IAM untuk Amazon GameLift.

Properti	Deskripsi
RoleArn	Amazon Resource Name (ARN) dari peran layanan yang memperluas akses terbatas ke sumber daya AndaAWS. Jenis: `string` Wajib: Ya
RoleSessionName	Nama sesi yang menjelaskan penggunaan kredensyal peran.

Properti	Deskripsi
	Tipe: `string` 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 `AttrType` enum. Wajib: Tidak
D	Merupakan nilai atribut string. Jenis: `string` Wajib: Ya
N	Merupakan nilai atribut numerik. Jenis: `double` Wajib: Ya
SL	Merupakan array nilai atribut string. Jenis: `string[]` Wajib: Ya
SDM	Merupakan kamus kunci string dan nilai ganda.

Properti	Deskripsi
	Jenis: `Dictionary<string, double>` Wajib: Ya

AwsStringOutcome

Tipe data ini hasil dari tindakan dan menghasilkan sebuah objek dengan sifat sebagai berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan. Tipe: `string` Wajib: Tidak
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: `bool` Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Tipe: the section called “GameLiftError” Wajib: Tidak

GenericOutcome

Tipe data ini hasil dari tindakan dan menghasilkan sebuah objek dengan sifat sebagai berikut:

Properti	Deskripsi
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: `bool`

Properti	Deskripsi
	Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Tipe: the section called “GameLiftError” Wajib: Tidak

DescribePlayerSessionsOutcome

Tipe data ini hasil dari tindakan dan menghasilkan sebuah objek dengan sifat sebagai berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan. Tipe: the section called “DescribePlayerSessionsResult” Wajib: Tidak
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: `bool` Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Tipe: the section called “GameLiftError” Wajib: Tidak

DescribePlayerSessionsResult

Properti	Deskripsi
NextToken	Token menunjukkan awal halaman berikutnya hasil. Untuk menentukan awal set hasil, jangan berikan nilai. Jika Anda memberikan ID sesi pemain, parameter ini akan diabaikan. Jenis: `string` Wajib: Ya
PlayerSessions	Kumpulan objek yang berisi properti untuk setiap sesi pemain yang cocok dengan permintaan. Tipe: `IList<`the section called “PlayerSession” `>` Diperlukan:
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: `bool` Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Tipe: the section called “GameLiftError” Wajib: Tidak

PlayerSession

Properti	Deskripsi
CreationTime	Jenis: `long`

Properti	Deskripsi
	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: Sebuah `PlayerSessionStatus` enum. Wajib: Ya
TerminationTime	Jenis: `long` Wajib: Ya

Properti	Deskripsi
DnsName	Jenis: `string` Wajib: Ya

StartMatchBackfillOutcome

Tipe data ini hasil dari tindakan dan menghasilkan sebuah objek dengan sifat sebagai berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan. Tipe: the section called “StartMatchBackfillResult” Wajib: Tidak
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: `bool` Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Tipe: the section called “GameLiftError” Wajib: Tidak

StartMatchBackfillResult

Properti	Deskripsi
TicketId	Jenis: `string` Wajib: Ya

GetComputeCertificateOutcome

Tipe data ini hasil dari tindakan dan menghasilkan sebuah objek dengan sifat sebagai berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan. Tipe: 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. Tipe: 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 hasil dari tindakan dan menghasilkan sebuah objek dengan sifat sebagai berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan. Tipe: 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. Tipe: the section called “GameLiftError” Wajib: Tidak

GetFleetRoleCredentialsResult

Properti	Deskripsi
AccessKeyId	ID kunci akses untuk mengautentikasi dan menyediakan akses ke AWS sumber daya Anda. Tipe: `string` Wajib: Tidak
AssumedRoleId	ID pengguna yang peran layanan milik.

Properti	Deskripsi
	Tipe: `string` Wajib: Tidak
AssumedRoleUserArn	Amazon Resource Name (ARN) pengguna yang menjadi milik peran layanan. Tipe: `string` Wajib: Tidak
Kedaluwarsa	Jumlah waktu hingga kredensyal sesi Anda kedaluwarsa. Tipe: `DateTime` Wajib: Tidak
SecretAccessKey	ID kunci akses rahasia untuk otentikasi. Tipe: `string` Wajib: Tidak
SessionToken	Token untuk mengidentifikasi sesi aktif saat ini yang berinteraksi dengan AWS sumber daya Anda. Tipe: `string` Wajib: Tidak
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: `bool` Wajib: Ya

Properti	Deskripsi
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Tipe: the section called “GameLiftError” Wajib: Tidak

AwsDateTimeOutcome

Tipe data ini hasil dari tindakan dan menghasilkan sebuah objek dengan sifat sebagai berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan. Tipe: `DateTime` Wajib: Tidak
Berhasil	Apakah tindakan itu berhasil atau tidak. Jenis: `bool` Wajib: Ya
Kesalahan	Kesalahan yang terjadi jika tindakan tidak berhasil. Tipe: the section called “GameLiftError” Wajib: Tidak

GameLiftError

Properti	Deskripsi
ErrorType	Jenis kesalahan.

Properti	Deskripsi
	Jenis: Sebuah `GameLiftErrorType` enum. Wajib: Tidak
ErrorName	Nama kesalahan. Tipe: `string` Wajib: Tidak
ErrorMessage	Pesan kesalahan. Tipe: `string` Wajib: Tidak

enum

Enum yang ditentukan untuk Amazon GameLift server SDK (C#) didefinisikan sebagai berikut:

AttrType

- TIDAK ADA
- STRING
- DOUBLE
- STRING_DAFTAR
- STRING_DOUBLE_MAP

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 ke Amazon gagal. GameLift
- NETWORK_NOT_INITIALIZED — Jaringan belum diinisialisasi.
- GAMESESSION_ID_NOT_SET - ID sesi game belum ditetapkan.
- BAD_REQUEST_EXCEPTION
- INTERNAL_SERVICE_EXCEPTION

- ALREADY_INITIALIZED — GameLift Server atau Klien Amazon telah diinisialisasi dengan Initialize ().
- FLEET_MISMATCH - Armada target tidak cocok dengan armada GameSession atau playerSession.
- GAMELIFT_CLIENT_NOT_INITIALIZED — Klien Amazon belum diinisialisasi. GameLift
- GAMELIFT_SERVER_NOT_INITIALIZED — Server Amazon belum diinisialisasi. GameLift
- GAME_SESSION_ENDED_FAILED — Amazon GameLift Server SDK tidak dapat menghubungi layanan untuk melaporkan sesi permainan berakhir.
- GAME_SESSION_NOT_READY — Sesi Game Server Amazon tidak diaktifkanGameLift.
- GAME_SESSION_READY_FAILED — Amazon GameLift Server SDK tidak dapat menghubungi layanan untuk melaporkan sesi game sudah siap.
- INITIALIZATION_MISMATCH - Metode klien dipanggil setelah Server: :Initialize (), atau sebaliknya.
- NOT_INITIALIZED — GameLift Server atau Klien Amazon belum diinisialisasi dengan Initialize ().
- NO_TARGET_ALIASID_SET - Target AliAsid belum ditetapkan.
- NO_TARGET_FLEET_SET - Armada target belum ditetapkan.
- PROCESS_ENDING_FAILED — Amazon GameLift Server SDK tidak dapat menghubungi layanan untuk melaporkan proses berakhir.
- PROCESS_NOT_ACTIVE - Proses server belum aktif, tidak terikat padaGameSession, dan tidak dapat menerima atau memproses. PlayerSessions
- PROCESS_NOT_READY - Proses server belum siap untuk diaktifkan.
- PROCESS_READY_FAILED — Amazon GameLift Server SDK tidak dapat menghubungi layanan untuk melaporkan proses sudah siap.
- SDK_VERSION_DETECTION_FAILED — Deteksi versi SDK gagal.
- STX_CALL_FAILED - Panggilan ke komponen backend server xStX telah gagal.
- STX_INITIALIZATION_FAILED - Komponen backend server xStX gagal diinisialisasi.
- 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 Retriable 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
- TIDAK_SET
- RESERVED
- TIMEDOUT

Referensi SDK 4.x GameLift server Amazon untuk C

Referensi Amazon GameLift C# Server SDK 4.x ini dapat membantu Anda mempersiapkan game multipemain untuk digunakan dengan Amazon. GameLift Untuk informasi detail tentang proses integrasi, lihat Tambahkan Amazon GameLift ke server game Anda.

Topik

- Referensi SDK GameLift server Amazon (C #): Tindakan
- Referensi SDK GameLift server Amazon (C #): Tipe data

Referensi SDK GameLift server Amazon (C #): Tindakan

Anda dapat menggunakan referensi SDK server Amazon GameLift C# ini untuk membantu Anda mempersiapkan game multipemain untuk digunakan dengan Amazon. GameLift Untuk detail tentang proses integrasi, lihatTambahkan Amazon GameLift ke server game Anda.

- Tindakan
- Jenis Data

AcceptPlayerSession()

Memberitahu GameLift layanan Amazon bahwa pemain dengan ID sesi pemain tertentu telah terhubung ke proses server dan memerlukan validasi. Amazon GameLift memverifikasi bahwa ID sesi pemain valid—yaitu, ID pemain telah memesan slot pemain dalam sesi permainan. Setelah divalidasi, Amazon GameLift mengubah status slot pemain dari RESERVED menjadi AKTIF.

Sintaks

```
GenericOutcome AcceptPlayerSession(String playerSessionId)
```

Parameter

playerSessionId

ID unik yang dikeluarkan oleh Amazon GameLift saat sesi pemain baru dibuat. ID sesi pemain ditentukan dalam `PlayerSession` objek, yang dikembalikan sebagai respons terhadap panggilan klien ke tindakan GameLiftAPI StartGameSessionPlacement,, CreateGameSessionDescribeGameSessionPlacement, atau DescribePlayerSessions.

Tipe: String

Wajib: Ya

Nilai kembali

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 ID sesi pemain yang tidak valid.

```
void ReceiveConnectingPlayerSessionID (Connection connection, String playerSessionId){
    var acceptPlayerSessionOutcome =
 GameLiftServerAPI.AcceptPlayerSession(playerSessionId);
     if(acceptPlayerSessionOutcome.Success)
    {
        connectionToSessionMap.emplace(connection, playerSessionId);
        connection.Accept();
    }
     else
    {
        connection.Reject(acceptPlayerSessionOutcome.Error.ErrorMessage);    }
}
```

ActivateGameSession()

Memberitahu GameLift layanan Amazon bahwa proses server telah mengaktifkan sesi permainan dan sekarang siap untuk 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 kembali

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 kembali

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 PlayerSessiondata AWS SDK Amazon GameLift API.

Contoh

Contoh ini menggambarkan permintaan untuk semua sesi pemain yang secara aktif terhubung ke sesi game tertentu. Dengan menghilangkan NextTokendan menetapkan nilai Limit ke 10, Amazon GameLift akan mengembalikan 10 rekaman sesi 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 kembali `Success` = `True` dan `GameSessionId` = `""` (string kosong).

Sintaks

```
AwsStringOutcome GetGameSessionId()
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai kembali

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-yang dikaitkan dengan armada dan instansinya. AWS Certificate Managermenghasilkan sertifikat ini ketika Anda membuat armada baru

dengan konfigurasi sertifikat diatur 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 kembali

Jika berhasil, mengembalikan `GetInstanceCertificateOutcome` objek yang berisi lokasi file sertifikat TLS armada dan rantai sertifikat, yang disimpan pada instance. File sertifikat root, yang diekstrak dari rantai sertifikat, juga disimpan pada instance. Jika tidak berhasil, ini mengembalikan pesan kesalahan.

Untuk informasi selengkapnya tentang data sertifikat dan 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 kembali

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()` callback dari GameLift layanan Amazon. Amazon GameLift dapat memanggil `onProcessTerminate()` karena alasan berikut: (1) untuk kesehatan yang buruk (proses server telah melaporkan kesehatan port atau belum merespons AmazonGameLift, (2) saat mengakhiri instans selama peristiwa penurunan skala, atau (3) saat instans dihentikan karena gangguan instans.

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 kembali

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 637355968000000000 detik. Jika tidak ada waktu penghentian tersedia, pesan kesalahan ditampilkan.

Contoh

```
var getTerminationTimeOutcome = GameLiftServerAPI.GetTerminationTime();
```

InitSDK()

Menginisialisasi Amazon GameLift SDK. Metode ini harus dipanggil pada peluncuran, sebelum inisialisasi GameLift terkait Amazon lainnya terjadi.

Sintaks

```
InitSDKOutcome InitSDK()
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai kembali

Jika berhasil, mengembalikan InitSdkOutcome objek yang menunjukkan bahwa proses server siap untuk memanggilProcessReady().

Contoh

```
var initSDKOutcome = GameLiftServerAPI.InitSDK();
```

ProcessEnding()

Memberitahu GameLift layanan Amazon 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 kembali

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 GameLift layanan Amazon bahwa proses server siap untuk menyelenggarakan sesi game. 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 GameLift layanan Amazon untuk berkomunikasi dengan proses server.
- Nomor port yang didengarkan oleh proses server.
- Jalur ke file khusus sesi game apa pun yang Anda GameLift ingin Amazon tangkap dan simpan.

Wajib: Ya

Nilai kembali

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 GameLift layanan Amazon bahwa pemain dengan ID sesi pemain tertentu telah terputus dari proses server. Sebagai tanggapan, Amazon GameLift mengubah slot pemain menjadi tersedia, yang memungkinkannya ditugaskan ke pemain baru.

Sintaks

```
GenericOutcome RemovePlayerSession(String playerSessionId)
```

Parameter

playerSessionId

ID unik yang dikeluarkan oleh Amazon GameLift saat sesi pemain baru dibuat. ID sesi pemain ditentukan dalam `PlayerSession` objek, yang dikembalikan sebagai respons terhadap panggilan klien ke tindakan GameLiftAPI StartGameSessionPlacement,, CreateGameSessionDescribeGameSessionPlacement, atau DescribePlayerSessions.

Tipe: String

Wajib: Ya

Nilai kembali

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 denganFlexMatch. Lihat juga tindakan 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, GameLift layanan Amazon mengirimkan 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

```
StartMatchBackfillOutcome StartMatchBackfill (StartMatchBackfillRequest
 startBackfillRequest);
```

Parameter

StartMatchBackfillRequest

Sebuah objek StartMatchBackfillRequest yang mengkomunikasikan informasi berikut:

- ID tiket untuk ditetapkan ke permintaan backfill. Informasi ini bersifat opsional; jika tidak ada ID yang disediakan, Amazon GameLift akan membuat satu secara otomatis.
- Matchmaker untuk dikirimi 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 kembali

Mengembalikan StartMatchBackfillOutcome objek dengan ID tiket pengurukan 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 tindakan 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 dikirimi permintaan backfill
- sesi game yang terkait dengan permintaan backfill

Wajib: Ya

Nilai kembali

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. Sebagai gantinya, proses server harus menelepon ProcessEnding() setelah sesi permainan berakhir.

Memberitahu GameLift layanan Amazon bahwa proses server telah mengakhiri sesi game 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 sinyal ke Amazon GameLift bahwa proses server segera tersedia untuk menjadi tuan rumah 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 kembali

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 Amazon GameLift Service API Reference).

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 kembali

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 updatePlayerSessionCreationPolicyOutcomex =

 GameLiftServerAPI.UpdatePlayerSessionCreationPolicy(PlayerSessionCreationPolicy.ACCEPT_ALL);
```

Referensi SDK GameLift server Amazon (C #): Tipe data

Anda dapat menggunakan referensi SDK server Amazon GameLift C# ini untuk membantu Anda mempersiapkan game multipemain untuk digunakan dengan Amazon. GameLift Untuk detail tentang proses integrasi, lihatTambahkan Amazon GameLift ke server game Anda.

- Aksi
- Jenis Data

LogParameters

Tipe data ini digunakan untuk mengidentifikasi file mana yang dihasilkan selama sesi permainan yang Anda inginkan Amazon GameLift untuk mengunggah dan menyimpan setelah sesi permainan berakhir. Informasi ini dikomunikasikan ke GameLift layanan Amazon dalam ProcessReady() panggilan.

Daftar Isi

logPaths

Daftar jalur direktori ke file log server game yang ingin disimpan Amazon GameLift untuk akses di masa mendatang. 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 mutlak. 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. Hal 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 tertentu. 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 NextTokenuntuk 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. ID Pemain ditentukan oleh developer. Lihat Hasilkan ID pemain.

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 GameLift layanan Amazon 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.

Tipe: Bilangan Bulat

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 dipanggil GameLift layanan Amazon untuk mengaktifkan sesi game baru. Amazon GameLift memanggil fungsi ini sebagai respons terhadap permintaan klien CreateGameSession. Fungsi callback mengambil GameSessionobjek (didefinisikan dalam Amazon GameLift Service API Reference).

Tipe: `void OnStartGameSessionDelegate(GameSession gameSession)`

Wajib: Ya

onProcessTerminate

Nama fungsi callback yang dipanggil GameLift layanan Amazon untuk memaksa proses server dimatikan. Setelah memanggil fungsi ini, Amazon GameLift 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 dipanggil GameLift layanan Amazon untuk meminta laporan status kesehatan dari proses server. Amazon GameLift memanggil fungsi ini setiap 60 detik. Setelah memanggil fungsi ini Amazon GameLift 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 dipanggil GameLift layanan Amazon untuk meneruskan objek sesi game yang diperbarui ke proses server. Amazon GameLift memanggil fungsi ini ketika permintaan isi ulang pertandingan telah diproses untuk menyediakan data mak comblang yang diperbarui. Ini melewati GameSessionobjek, 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 ke GameLift layanan Amazon dalam StartMatchBackfill() panggilan.

Daftar Isi

GameSessionArn

Pengidentifikasi sesi game yang unik. Metode SDK GetGameSessionId() mengembalikan identifier dalam format ARN.

Tipe: String

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.

Tipe: String

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 Panduan Referensi GameLift API Amazon untuk deskripsi format objek Player. Untuk menemukan atribut pemain, ID, dan tugas tim, lihat di objek sesi game, di properti data matchmaker. 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 disediakan di sini, Amazon GameLift 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 ke GameLift layanan Amazon dalam StopMatchBackfill() panggilan.

Daftar Isi

GameSessionArn

Pengidentifikasi sesi game unik yang terkait dengan permintaan yang dibatalkan.

Tipe: String

Wajib: Ya

MatchmakingConfigurationArn

Pengidentifikasi unik dari matchmaker sebagai tujuan pengiriman permintaan ini.

Tipe: String

Wajib: Ya

TicketId

Pengidentifikasi unik dari tiket backfill match yang akan dibatalkan.

Tipe: String

Wajib: Ya

Referensi SDK GameLift server Amazon untuk Go

Anda dapat menggunakan referensi SDK server Amazon GameLift Go ini untuk membantu Anda mempersiapkan game multipemain untuk digunakan dengan AmazonGameLift. Untuk detail tentang proses integrasi, lihatTambahkan Amazon GameLift ke server game Anda.

Topik

- Referensi SDK GameLift server Amazon (Go): Tindakan
- Referensi SDK GameLift server Amazon (Go): Tipe data

Referensi SDK GameLift server Amazon (Go): Tindakan

Anda dapat menggunakan referensi SDK server Amazon GameLift Go ini untuk membantu Anda mempersiapkan game multipemain untuk digunakan dengan Amazon. GameLift Untuk detail tentang proses integrasi, lihatTambahkan Amazon GameLift ke server game Anda.

`GameLiftServerAPI.go`mendefinisikan tindakan SDK server Go.

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 SDK. Panggil metode ini saat diluncurkan sebelum inisialisasi lain yang terkait dengan Amazon GameLift terjadi. Metode ini mengatur komunikasi antara server dan GameLift layanan Amazon.

Sintaksis

```
func InitSDK(params ServerParameters) error
```

Parameter-parameter

ServerParameters

Untuk menginisialisasi server game di GameLift Anywhere armada Amazon, 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 GameLift armada Amazon yang berisi GameLift Anywhere komputasi Amazon Anda.
- Token otorisasi yang dihasilkan oleh GameLift operasi Amazon.

Untuk menginisialisasi server game pada armada EC2 yang GameLift dikelola Amazon, buat `ServerParameters` objek tanpa parameter. Dengan panggilan ini, GameLift agen Amazon menyiapkan lingkungan komputasi dan secara otomatis terhubung ke GameLift layanan Amazon untuk Anda.

Nilai yang ditampilkan

Jika berhasil, mengembalikan `nil` kesalahan untuk menunjukkan bahwa proses server siap untuk memanggilProcessReady().

> **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

GameLift AnywhereContoh Amazon

```
//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 GameLift agent to enable
 further communication.
err := server.InitSDK(serverParameters)
```

Contoh EC2 yang GameLift dikelola Amazon

```
//Define the server parameters
serverParameters := ServerParameters {}

//Call InitSDK to establish a local connection with the GameLift agent to enable
 further communication.
err := server.InitSDK(serverParameters)
```

ProcessReady()

Memberitahu Amazon GameLift bahwa proses server siap untuk meng-host sesi game. Panggil metode ini setelah memanggilInitSDK(). Metode ini harus dipanggil hanya satu kali per proses.

Sintaksis

```
func ProcessReady(param ProcessParameters) error
```

Parameter-parameter

ProcessParameters

ProcessParametersObjek mengkomunikasikan informasi berikut tentang proses server:

- Nama-nama metode callback yang diterapkan dalam kode server game yang dipanggil GameLift layanan Amazon 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 Anda GameLift ingin Amazon tangkap dan simpan.

Nilai yang ditampilkan

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 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 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.

Sintaksis

```
func UpdatePlayerSessionCreationPolicy(policy model.PlayerSessionCreationPolicy) error
```

Parameter-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 ditampilkan

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.

Sintaksis

```
func GetGameSessionID() (string, error)
```

Parameter-parameter

Tindakan ini tidak memiliki parameter.

Nilai yang ditampilkan

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. Amazon GameLift menyerukan `onProcessTerminate()` alasan berikut:

- Ketika proses server telah melaporkan kesehatan yang buruk atau belum menanggapi Amazon GameLift.
- 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()

Memberi tahu Amazon GameLift bahwa pemain dengan ID sesi pemain yang ditentukan telah terhubung ke proses server dan memerlukan validasi. Amazon GameLift memverifikasi bahwa ID sesi pemain valid. Setelah sesi pemain divalidasi, Amazon GameLift mengubah status slot pemain dari `RESERVED` ke`ACTIVE`.

Sintaksis

```
func AcceptPlayerSession(playerSessionID string) error
```

Parameter-parameter

`playerSessionId`

ID unik yang dikeluarkan oleh Amazon GameLift saat sesi pemain baru dibuat.

Nilai yang ditampilkan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Contoh ini menangani permintaan koneksi yang mencakup memvalidasi dan menolak ID sesi pemain yang tidak valid.

```
func ReceiveConnectingPlayerSessionID(conn Connection, playerSessionID string) {
    err := server.AcceptPlayerSession(playerSessionID)
    if err != nil {
        connection.Accept()
    } else {
        connection.Reject(err.Error())
    }
}
```

RemovePlayerSession()

Memberitahu Amazon GameLift bahwa pemain telah terputus dari proses server. Sebagai tanggapan, Amazon GameLift mengubah slot pemain menjadi tersedia.

Sintaksis

```
func RemovePlayerSession(playerSessionID string) error
```

Parameter-parameter

`playerSessionId`

ID unik yang dikeluarkan oleh Amazon GameLift saat sesi pemain baru dibuat.

Nilai yang ditampilkan

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

Sintaksis

```
func DescribePlayerSessions(req request.DescribePlayerSessionsRequest)
 (result.DescribePlayerSessionsResult, error) {
 return srv.describePlayerSessions(&req)
}
```

Parameter-parameter

DescribePlayerSessionsRequest

Sebuah `DescribePlayerSessionsRequest` objek menjelaskan sesi pemain mana yang akan diambil.

Nilai yang ditampilkan

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 NextTokendan menyetel nilai Batas ke 10, Amazon GameLift 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, Amazon GameLift 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.

Sintaksis

```
func StartMatchBackfill(req request.StartMatchBackfillRequest)
 (result.StartMatchBackfillResult, error)
```

Parameter-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 menghasilkannya.
- Matchmaker untuk dikirimi 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 ditampilkan

Mengembalikan `StartMatchBackfillResult` objek dengan ID tiket isi ulang pertandingan, 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.

Sintaksis

```
func StopMatchBackfill(req request.StopMatchBackfillRequest) error
```

Parameter-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 ditampilkan

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 GameLift Anywhere armada Amazon. Untuk informasi lebih lanjut, 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 GameLift terkelola Amazon, 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 kredensyal peran layanan yang Anda buat untuk memperluas izin ke orang lain ke Amazon. Layanan AWS GameLift Kredensyal ini memungkinkan server game Anda menggunakan sumber daya AndaAWS. Untuk informasi selengkapnya, lihat Menyiapkan peran layanan IAM untuk Amazon GameLift.

Sintaks

```
func GetFleetRoleCredentials(
  req request.GetFleetRoleCredentialsRequest,
) (result.GetFleetRoleCredentialsResult, error) {
  return srv.getFleetRoleCredentials(&req)
```

```
}
```

Parameter-parameter

GetFleetRoleCredentialsRequest

Kredensyal peran yang memperluas akses terbatas ke AWS sumber daya Anda ke server game.

Nilai yang ditampilkan

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 kredensyal 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 GameLift game Amazon 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, hubungi `Destroy()` lalu `InitSDK()` inisialisasi ulang sebelum memberi tahu Amazon GameLift bahwa prosesnya siap untuk meng-host 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)
  }
}
```

Referensi SDK GameLift server Amazon (Go): Tipe data

Anda dapat menggunakan referensi SDK server Amazon GameLift Go ini untuk membantu Anda mempersiapkan game multipemain untuk digunakan dengan Amazon. GameLift Untuk detail tentang proses integrasi, lihatTambahkan Amazon GameLift ke server game Anda.

Tipe Data

- LogParameters
- ProcessParameters
- UpdateGameSession
- GameSession
- ServerParameters
- StartMatchBackfillRequest
- Pemain
- DescribePlayerSessionsRequest

- StopMatchBackfillRequest
- GetFleetRoleCredentialsRequest

LogParameters

Objek yang mengidentifikasi file yang dihasilkan selama sesi permainan yang Anda GameLift ingin Amazon unggah dan simpan setelah sesi permainan berakhir. Server game menyediakan `LogParameters` ke Amazon GameLift sebagai bagian dari `ProcessParameters` objek dalam ProcessReady() panggilan.

Sifat-sifat	Deskripsi
`LogPaths`	Daftar jalur direktori ke file log server game yang Anda ingin Amazon simpan GameLift untuk akses masa depan. 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: `[]string` Wajib: Tidak

ProcessParameters

Objek yang menggambarkan komunikasi antara proses server dan Amazon GameLift. Proses server memberikan informasi ini ke Amazon GameLift dengan panggilan keProcessReady().

Sifat-sifat	Deskripsi
`LogParameters`	Objek dengan jalur direktori ke file yang dihasilkan selama sesi permainan. Amazon GameLift menyalin dan menyimpan file untuk akses future. Jenis: `LogParameters` Wajib: Tidak

`OnHealthCheck`	Fungsi callback yang GameLift dipanggil Amazon untuk meminta laporan status kesehatan dari proses server. Amazon GameLift 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 mencatat proses server sebagai tidak sehat. Jenis: `OnHealthCheck func() bool` Wajib: Tidak
`OnProcessTerminate`	Fungsi callback yang GameLift dipanggil Amazon untuk memaksa proses server dimatikan. Setelah memanggil fungsi ini, Amazon GameLift menunggu 5 menit hingga proses server dimatikan dan merespons dengan ProcessEnding() panggilan sebelum mematikan proses server. Jenis: `OnProcessTerminate func()` Wajib: Ya
`OnStartGameSession`	Fungsi callback yang GameLift dipanggil Amazon untuk meneruskan objek sesi game yang diperbarui ke proses server. Amazon GameLift memanggil fungsi ini ketika permintaan pengisian ulang kecocokan telah diproses untuk menyediakan data mak comblang yang diperbarui. Ini melewati GameSessionobjek, pembaruan status (`updateReason`), dan ID tiket isi ulang pertandingan. Jenis: `OnStartGameSession func (model.GameSession )` Wajib: Ya
`OnUpdateGameSession`	Fungsi callback yang GameLift dipanggil Amazon untuk meneruskan informasi sesi game yang diperbarui ke proses server. Amazon GameLift memanggil fungsi ini setelah memproses permintaan pengisian ulang kecocokan untuk menyediakan data mak comblang yang diperbarui. Jenis: `OnUpdateGameSession func (model.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: `int` 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	GameSessionObjek yang ditentukan oleh Amazon GameLift API. `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: Tidak

GameSession

Detail sesi permainan.

Properti	Deskripsi
GameSessionId	Pengenal unik untuk sesi permainan. Sesi permainan Nama Sumber Daya Amazon (ARN) memiliki format berikut:. `arn:aws:gamelift:<region>::gamesession/<fleet ID>/<custom ID string or idempotency token>` 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
MaximumPlayerSessionCount	Jumlah maksimum koneksi pemain ke sesi permainan. Jenis: `Integer` Wajib: Tidak
Port	Nomor port untuk sesi permainan. Untuk terhubung ke server GameLift game Amazon, aplikasi memerlukan alamat IP dan nomor port. Jenis: `Integer` Wajib: Tidak
IpAddress	Alamat IP dari sesi game. Untuk terhubung ke server GameLift game Amazon, aplikasi memerlukan alamat IP dan nomor port.

Properti	Deskripsi
	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
GameProperties	Satu set properti kustom untuk sesi permainan, diformat sebagai pasangan key:value. Properti ini diteruskan dengan permintaan untuk memulai sesi permainan baru. Jenis: `map[string] string` 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` . • Armada yang tidak mendukung TLS:`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 GameLift Anywhere server Amazon dan GameLift layanan Amazon. Informasi ini digunakan saat meluncurkan proses server baru dengan[InitSDK()](). Untuk server yang dihosting di instans EC2 GameLift terkelola Amazon, gunakan objek kosong.

Properti	Deskripsi
`WebSocketURL`	`GameLiftServerSdkEndpoint` Amazon GameLift kembali ketika Anda `RegisterCompute` untuk sumber daya GameLift Anywhere komputasi Amazon. Jenis: `string` Wajib: Ya
`ProcessID`	Pengenal unik yang terdaftar pada proses server yang menghosting game Anda. Jenis: `string`

Properti	Deskripsi
	Wajib: Ya
`HostID`	Pengidentifikasi unik dari sumber daya komputasi yang menghosting proses server baru. `HostID`Ini adalah yang `ComputeName` digunakan saat Anda mendaftarkan komputasi Anda. Untuk informasi lebih lanjut, lihat RegisterCompute. Jenis: `string` Wajib: Ya
`FleetID`	Pengidentifikasi unik armada tempat komputasi terdaftar. Untuk informasi lebih lanjut, lihat RegisterCompute. Jenis: `string` Wajib: Ya
`AuthToken`	Token otentikasi yang dihasilkan oleh Amazon GameLift yang mengauten tikasi server Anda ke Amazon. GameLift Untuk informasi lebih lanjut, lihat GetComputeAuthToken. Jenis: `string` Wajib: Ya

StartMatchBackfillRequest

Informasi yang digunakan untuk membuat permintaan pengisian ulang perjodohan. Server game mengkomunikasikan informasi ini ke Amazon GameLift dalam satu StartMatchBackfill() panggilan.

Properti	Deskripsi
GameSessionArn	Pengidentifikasi sesi permainan yang unik. Operasi API `GetGameSessionId` mengembalikan pengenal dalam format ARN. Jenis: `String`

Properti	Deskripsi
	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: `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: `[]model.Player` Wajib: Ya
TicketId	Pengenal unik untuk tiket permintaan pencocokan atau pencocokan isi ulang. Jika Anda tidak memberikan nilai, Amazon GameLift menghasilkannya. Gunakan pengidentifikasi ini untuk melacak status tiket backfill match atau membatalkan permintaan jika diperlukan. Jenis: `String` Wajib: Tidak

Pemain

Objek yang mewakili pemain dalam perjodohan. Ketika permintaan perjodohan dimulai, pemain memiliki ID pemain, atribut, dan mungkin data latensi. Amazon GameLift 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: `map[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. Untuk informasi selengkapnya tentang atribut pemain, lihat AttributeValue. Jenis: `map[string] AttributeValue` 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

Objek yang menentukan sesi pemain mana yang akan diambil. Proses server menyediakan informasi ini dengan DescribePlayerSessions() panggilan ke Amazon GameLift.

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`arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string>` . `GameSessionID` Ini adalah string ID kustom atau string yang dihasilkan. Jenis: `String` Wajib: Tidak
`PlayerSessionID`	Pengenal unik untuk sesi pemain. Gunakan parameter ini untuk meminta satu sesi pemain tertentu. Jenis: `String` Wajib: Tidak
`PlayerID`	Pengenal unik untuk pemain. Gunakan parameter ini untuk meminta semua sesi pemain untuk pemain tertentu. Lihat Hasilkan ID pemain. Jenis: `String` 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: `String`

Properti	Deskripsi
	Wajib: Tidak
`NextToken`	Token yang menunjukkan awal halaman hasil berikutnya. Untuk menentuka n awal kumpulan hasil, jangan berikan nilai. Jika Anda memberikan ID sesi pemain, parameter ini diabaikan. Jenis: `String` Wajib: Tidak
`Limit`	Jumlah hasil maksimum yang akan dikembalikan. Jika Anda memberikan ID sesi pemain, parameter ini diabaikan. Jenis: `int` Wajib: Tidak

StopMatchBackfillRequest

Informasi yang digunakan untuk membatalkan permintaan pengisian ulang perjodohan. Server game mengkomunikasikan informasi ini ke GameLift layanan Amazon dalam StopMatchBackfill() panggilan.

Properti	Deskripsi
GameSessi onArn	Pengidentifikasi sesi permainan unik dari permintaan yang dibatalkan. Jenis: `string` Wajib: Tidak
Matchmaki ngConfigu rationArn	Pengidentifikasi unik dari mak comblang permintaan ini dikirim ke. Jenis: `string` Wajib: Tidak
TicketId	Pengenal unik dari tiket permintaan isi ulang yang akan dibatalkan. Jenis: `string`

Properti	Deskripsi
	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.

Properti	Deskripsi
RoleArn	ARN dari peran layanan yang memperluas akses terbatas ke sumber daya Anda. AWS Jenis: `string` Wajib: Ya
RoleSessionName	Nama sesi yang menjelaskan penggunaan kredensi peran. Jenis: `string` Wajib: Ya

Referensi SDK GameLift server Amazon untuk Unreal Engine

Referensi Amazon GameLift Server SDK ini dapat membantu Anda mempersiapkan proyek game Unreal Engine Anda untuk digunakan dengan Amazon. GameLift Untuk informasi detail tentang proses integrasi, lihat Tambahkan Amazon GameLift ke server game Anda.

API ini didefinisikan dalam `GameLiftServerSDK.h` dan `GameLiftServerSDKModels.h`.

Untuk menyiapkan plugin Unreal Engine dan melihat contoh kode Integrasikan Amazon GameLift ke dalam proyek Unreal Engine.

Topik

- Referensi SDK GameLift 5.x server Amazon Unreal Engine
- Referensi SDK 3.x server Amazon GameLift Unreal Engine

Referensi SDK GameLift 5.x server Amazon Unreal Engine

Anda dapat menggunakan referensi SDK 5.x server Amazon GameLift Unreal Engine ini untuk membantu Anda mempersiapkan game multipemain untuk digunakan dengan Amazon. GameLift Untuk detail tentang proses integrasi, lihatTambahkan Amazon GameLift ke server game Anda, dan untuk informasi tentang penggunaan plugin server Unreal SDK, lihat. Integrasikan Amazon GameLift ke dalam proyek Unreal Engine

Topik

- Referensi 5.x SDK GameLift server Amazon (Tidak nyata): Tindakan
- Referensi SDK GameLift server Amazon (Unreal): Tipe data

Referensi 5.x SDK GameLift server Amazon (Tidak nyata): Tindakan

Anda dapat menggunakan referensi SDK server Amazon GameLift Unreal ini untuk membantu Anda mempersiapkan game multipemain untuk digunakan dengan Amazon. GameLift Untuk detail tentang proses integrasi, lihat Tambahkan Amazon GameLift ke server game Anda dan untuk informasi tentang penggunaan plugin Unreal SDK server, lihat. Integrasikan Amazon GameLift ke dalam proyek Unreal Engine

Tindakan

- GetSdkVersion()
- InitSDK()
- InitSDK()
- ProcessReady()
- ProcessEnding()
- ActivateGameSession()
- UpdatePlayerSessionCreationPolicy()
- GetGameSessionId()
- GetTerminationTime()
- AcceptPlayerSession()
- RemovePlayerSession()
- DescribePlayerSessions()

- StartMatchBackfill()
- StopMatchBackfill()
- GetComputeCertificate()
- GetFleetRoleCredentials()

> **Note**
>
> Topik ini menjelaskan Amazon GameLift 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.

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 “F GameLiftStringOutcome”. 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 SDK untuk armada EC2 terkelola. Panggil metode ini saat peluncuran, sebelum inisialisasi lain yang terkait dengan Amazon GameLift terjadi. Metode ini membaca parameter server dari lingkungan host untuk mengatur komunikasi antara server dan GameLift layanan Amazon.

Sintaks

```
FGameLiftGenericOutcome InitSDK()
```

Nilai yang ditampilkan

Jika berhasil, mengembalikan `InitSdkOutcome` objek yang menunjukkan bahwa proses server siap untuk memanggilProcessReady().

Contoh

```
//Call InitSDK to establish a local connection with the GameLift agent to enable
 further communication.
FGameLiftGenericOutcome initSdkOutcome = gameLiftSdkModule->InitSDK();
```

InitSDK()

Menginisialisasi Amazon GameLift SDK untuk armadaAnywhere. Panggil metode ini saat peluncuran, sebelum inisialisasi lain yang terkait dengan Amazon GameLift terjadi. Metode ini memerlukan parameter server eksplisit untuk mengatur komunikasi antara server dan GameLift layanan Amazon.

Sintaksis

```
FGameLiftGenericOutcome InitSDK(serverParameters)
```

Parameter-parameter

F ServerParameters

Untuk menginisialisasi server game di GameLift Anywhere armada Amazon, 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 GameLift armada Amazon yang berisi GameLift Anywhere komputasi Amazon Anda.
- Token otorisasi yang dihasilkan oleh GameLift operasi Amazon.

Nilai yang ditampilkan

Jika berhasil, mengembalikan `InitSdkOutcome` objek yang menunjukkan bahwa proses server siap untuk memanggilProcessReady().

> **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 GameLift agent to enable
 further communication.
FGameLiftGenericOutcome initSdkOutcome = gameLiftSdkModule->InitSDK(serverParameters);
```

ProcessReady()

Memberitahu Amazon GameLift bahwa proses server siap untuk meng-host sesi game. Panggil metode ini setelah memanggilInitSDK(). Metode ini harus dipanggil hanya satu kali per proses.

Sintaksis

```
GenericOutcome ProcessReady(const Aws::GameLift::Server::ProcessParameters
&processParameters);
```

Parameter-parameter

processParameters

F ProcessParametersObjek yang mengkomunikasikan informasi berikut tentang proses server:

- Nama metode callback yang diterapkan dalam kode server game yang dipanggil GameLift layanan Amazon untuk berkomunikasi dengan proses server.
- Nomor port yang didengarkan oleh proses server.
- Jalur ke file khusus sesi game apa pun yang Anda GameLift ingin Amazon tangkap dan simpan.

Nilai yang ditampilkan

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 GameLift this game server is ready to receive incoming
 game sessions!
UE_LOG(GameServerLog, Log, TEXT("Calling Process Ready"));
FGameLiftGenericOutcome processReadyOutcome = gameLiftSdkModule-
>ProcessReady(*params);
```

ProcessEnding()

Memberi tahu Amazon GameLift 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. GameLift 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 GameLift 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 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, GameLift 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.

Sintaksis

```
FGameLiftGenericOutcome UpdatePlayerSessionCreationPolicy(EPlayerSessionCreationPolicy
 policy)
```

Parameter-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 ditampilkan

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 “F GameLiftError”

Sintaksis

```
FGameLiftStringOutcome GetGameSessionId()
```

Parameter-parameter

Tindakan ini tidak memiliki parameter.

Nilai yang ditampilkan

Jika berhasil, ini mengembalikan ID sesi game sebagai objek the section called “F GameLiftStringOutcome”. 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, GameLift sends an activation request to the game
 server and passes along the game session object containing game properties and other
 settings.
//Here is where a game server should take action based on the game session object.
//Once the game server is ready to receive incoming player connections, it should
 invoke GameLiftServerAPI.ActivateGameSession()
auto onGameSession = [=](Aws::GameLift::Server::Model::GameSession gameSession)
{
  FString gameSessionId = FString(gameSession.GetGameSessionId());
  UE_LOG(GameServerLog, Log, TEXT("GameSession Initializing: %s"), *gameSessionId);
  gameLiftSdkModule->ActivateGameSession();
};
```

GetTerminationTime()

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. Amazon GameLift `onProcessTerminate()` menyerukan alasan-alasan berikut:

- Ketika proses server telah melaporkan kesehatan yang buruk atau belum menanggapi Amazon GameLift.
- 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. `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.

Jika proses belum menerima `ProcessParameters.OnProcessTerminate()` panggilan balik, pesan kesalahan akan dikembalikan. Untuk informasi selengkapnya tentang mematikan proses server, lihatMenanggapi notifikasi shutdown proses server.

Contoh

```
AwsDateTimeOutcome TermTimeOutcome = gameLiftSdkModule->GetTerminationTime();
```

AcceptPlayerSession()

Memberitahu Amazon GameLift bahwa pemain dengan ID sesi pemain tertentu telah terhubung ke proses server dan perlu validasi. Amazon GameLift memverifikasi bahwa ID sesi pemain valid. Setelah sesi pemain divalidasi, Amazon GameLift mengubah status slot pemain dari RESERVED menjadi AKTIF.

Sintaksis

```
FGameLiftGenericOutcome AcceptPlayerSession(const FString& playerSessionId)
```

Parameter-parameter

playerSessionId

ID unik yang dikeluarkan oleh Amazon GameLift saat sesi pemain baru dibuat.

Nilai yang ditampilkan

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Contoh ini menangani permintaan koneksi yang mencakup memvalidasi dan menolak ID sesi pemain yang tidak valid.

```
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 bahwa pemain telah terputus dari proses server. Sebagai tanggapan, Amazon GameLift mengubah slot pemain menjadi tersedia.

Sintaksis

```
FGameLiftGenericOutcome RemovePlayerSession(const FString& playerSessionId)
```

Parameter-parameter

`playerSessionId`

ID unik yang dikeluarkan oleh Amazon GameLift saat sesi pemain baru dibuat.

Nilai yang ditampilkan

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

Sintaksis

```
FGameLiftDescribePlayerSessionsOutcome DescribePlayerSessions(const
 FGameLiftDescribePlayerSessionsRequest &describePlayerSessionsRequest)
```

Parameter-parameter

F GameLiftDescribePlayerSessionsRequest

the section called “F GameLiftDescribePlayerSessionsRequest”Objek yang menggambarkan sesi pemain mana yang akan diambil.

Nilai yang ditampilkan

Jika berhasil, ini mengembalikan objek the section called “F GameLiftDescribePlayerSessionsOutcome” 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 NextTokendan menyetel nilai Batas ke 10, Amazon GameLift 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, Amazon GameLift 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.

Sintaksis

```
FGameLiftStringOutcome StartMatchBackfill (FStartMatchBackfillRequest
 &startBackfillRequest);
```

Parameter-parameter

F 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 akan menghasilkannya.
- Matchmaker untuk dikirimi 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 ditampilkan

Mengembalikan `StartMatchBackfillOutcome` objek dengan ID tiket isi ulang pertandingan, 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.

Sintaksis

```
FGameLiftGenericOutcome StopMatchBackfill (FStopMatchBackfillRequest
 &stopBackfillRequest);
```

Parameter-parameter

F 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 yang terkait dengan permintaan isi ulang.

Nilai yang ditampilkan

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 GameLift Anywhere sumber daya komputasi Amazon Anda dan Amazon. GameLift Anda dapat

menggunakan jalur sertifikat saat mendaftarkan perangkat komputasi ke GameLift Anywhere armada Amazon. Untuk informasi lebih lanjut 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_TO_TCHAR(outres.GetCertificatePath());
    result.m_computeName = UTF8_TO_TCHAR(outres.GetComputeName());
    return FGameLiftGetComputeCertificateOutcome(result);
  }
  else {
    return FGameLiftGetComputeCertificateOutcome(FGameLiftError(outcome.GetError()));
  }
  #else
  return FGameLiftGetComputeCertificateOutcome(FGameLiftGetComputeCertificateResult());
  #endif
}
```

GetFleetRoleCredentials()

Mengambil kredensyal peran IAM yang mengizinkan Amazon GameLift untuk berinteraksi dengan orang lain. Layanan AWS Untuk informasi selengkapnya, lihat Berkomunikasi dengan sumber daya AWS lain dari armada.

Sintaks

```
FGameLiftGetFleetRoleCredentialsOutcome
 FGameLiftServerSDKModule::GetFleetRoleCredentials(const
 FGameLiftGetFleetRoleCredentialsRequest &request)
```

Parameter-parameter

F GameLiftGetFleetRoleCredentialsRequest

Nilai yang ditampilkan

Mengembalikan objek the section called “F GameLiftGetFleetRoleCredentialsOutcome”.

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_TO_TCHAR(outres.GetAssumedUserRoleArn());
    result.m_assumedRoleId = UTF8_TO_TCHAR(outres.GetAssumedRoleId());
    result.m_accessKeyId = UTF8_TO_TCHAR(outres.GetAccessKeyId());
    result.m_secretAccessKey = UTF8_TO_TCHAR(outres.GetSecretAccessKey());
    result.m_sessionToken = UTF8_TO_TCHAR(outres.GetSessionToken());
    result.m_expiration = FDateTime::FromUnixTimestamp(outres.GetExpiration());
    return FGameLiftGetFleetRoleCredentialsOutcome(result);
  }
  else {
    return FGameLiftGetFleetRoleCredentialsOutcome(FGameLiftError(outcome.GetError()));
  }
  #else
  return
 FGameLiftGetFleetRoleCredentialsOutcome(FGameLiftGetFleetRoleCredentialsResult());
```

```
  #endif
}
```

Referensi SDK GameLift server Amazon (Unreal): Tipe data

Anda dapat menggunakan referensi SDK server Amazon GameLift Unreal ini untuk membantu Anda mempersiapkan game multipemain untuk digunakan dengan Amazon. GameLift Untuk detail tentang proses integrasi, lihat Tambahkan Amazon GameLift ke server game Anda dan untuk informasi tentang penggunaan plugin Unreal SDK server, lihat. Integrasikan Amazon GameLift ke dalam proyek Unreal Engine

Tipe Data

- F ProcessParameters
- UpdateGameSession
- GameSession
- F ServerParameters
- F StartMatchBackfillRequest
- FPlayer
- F GameLiftDescribePlayerSessionsRequest
- F StopMatchBackfillRequest
- F AttributeValue
- F GameLiftGetFleetRoleCredentialsRequest
- F GameLiftLongOutcome
- F GameLiftStringOutcome
- F GameLiftDescribePlayerSessionsOutcome
- F GameLiftDescribePlayerSessionsResult
- F GenericOutcome
- F GameLiftPlayerSession
- F GameLiftGetComputeCertificateOutcome
- F GameLiftGetComputeCertificateResult
- F GameLiftGetFleetRoleCredentialsOutcome
- F GetFleetRoleCredentialsResult
- F GameLiftError

- Enum

> **Note**
>
> Topik ini menjelaskan Amazon GameLift 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.

F ProcessParameters

Tipe data ini berisi kumpulan parameter yang dikirim ke Amazon GameLift dalam fileProcessReady().

Sifat-sifat	Deskripsi
LogParameters	Objek dengan jalur direktori ke file yang dihasilkan selama sesi permainan. Amazon GameLift menyalin dan menyimpan file untuk akses future. Jenis: `TArray<FString>` Wajib: Tidak
OnHealthCheck	Fungsi callback yang GameLift dipanggil Amazon untuk meminta laporan status kesehatan dari proses server. Amazon GameLift 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 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 GameLift dipanggil Amazon untuk memaksa proses server dimatikan. Setelah memanggil fungsi ini, Amazon GameLift menunggu 5 menit hingga proses server dimatikan dan merespons dengan ProcessEnding() panggilan sebelum mematikan proses server. Jenis: `FSimpleDelegate` Wajib: Ya
OnStartGameSession	Fungsi callback yang GameLift dipanggil Amazon untuk mengaktifkan sesi permainan baru. Amazon GameLift memanggil fungsi ini sebagai tanggapan atas permintaan klien CreateGameSession. Fungsi callback meneruskan GameSessionobjek, seperti yang didefinisikan dalam Referensi Amazon GameLift API. Properti ini adalah fungsi delegasi didefinisikan sebagai `DECLARE_DELEGATE_OneParam(FOnStartGameSession, Aws::GameLift::Server::Model::GameSession);` Jenis: `FOnStartGameSession` Wajib: Ya

OnUpdateGameSession	Fungsi callback yang GameLift dipanggil Amazon untuk meneruskan objek sesi game yang diperbarui ke proses server. Amazon GameLift memanggil fungsi ini ketika permintaan pengisian ulang kecocokan telah diproses untuk menyediakan data mak comblang yang diperbarui. Ini melewati GameSessionobjek, 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	GameSessionObjek yang ditentukan oleh Amazon GameLift API. `GameSession` Objek berisi properti yang menggambarkan sesi permainan. Jenis: `Aws::GameLift::Server::GameSession` Wajib: Tidak
UpdateReason	Alasan bahwa sesi permainan sedang diperbarui. Jenis: `enum class UpdateReason` • MATCHMAKING_DATA_DIPERBARUI • BACKFILL_FAILED • BACKFILL_TIMED_OUT • BACKFILL_CANCELLED Wajib: Tidak
BackfillTicketId	ID tiket isi ulang yang mencoba memperbarui sesi permainan. Jenis: `char[]` Wajib: Tidak

GameSession

Tipe data ini memberikan 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: `char[]` Wajib: Tidak
Nama	Label deskriptif dari sesi permainan. Jenis: `char[]` Wajib: Tidak
FleetId	Pengenal unik untuk armada tempat sesi permainan berjalan. Jenis: `char[]` Wajib: Tidak
MaximumPlayerSessionCount	Jumlah maksimum koneksi pemain ke sesi permainan. Jenis: `int` Wajib: Tidak
Port	Nomor port untuk sesi permainan. Untuk terhubung ke server GameLift game Amazon, aplikasi memerlukan alamat IP dan nomor port. Jenis: `int` Wajib: Tidak

Properti	Deskripsi
IpAddress	Alamat IP dari sesi game. Untuk terhubung ke server GameLift game Amazon, aplikasi memerlukan alamat IP dan nomor port. Jenis: `char[]` Wajib: Tidak
GameSessionData	Set properti sesi game khusus, diformat sebagai nilai string tunggal. Jenis: `char[]` 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: `char[]` 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: `GameProperty[]` 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` . • Armada yang tidak mendukung TLS:`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

F ServerParameters

Informasi yang digunakan untuk menjaga koneksi antara GameLift Anywhere server Amazon dan GameLift layanan Amazon. Informasi ini digunakan saat meluncurkan proses server baru dengan InitSDK(). Untuk server yang dihosting di instans EC2 GameLift terkelola Amazon, gunakan objek kosong.

Properti	Deskripsi
webSocketUrl	`GameLiftServerSdkEndpoint` Amazon GameLift kembali ketika Anda `RegisterCompute` untuk sumber daya GameLift Anywhere komputasi Amazon. Jenis: `char[]`

Properti	Deskripsi
	Wajib: Ya
ProsesSid	Pengenal unik yang terdaftar pada proses server yang menghosting game Anda. Jenis: `char[]` Wajib: Ya
hostId	`HostID`Ini adalah yang `ComputeName` digunakan saat Anda mendaftarkan komputasi Anda. Untuk informasi lebih lanjut lihat, RegisterCompute. Jenis: `char[]` Wajib: Ya
FleeTID	Pengidentifikasi unik armada tempat komputasi terdaftar. Untuk informasi lebih lanjut lihat, RegisterCompute. Jenis: `char[]` Wajib: Ya
AuthToken	Token otentikasi yang dihasilkan oleh Amazon GameLift yang mengautentikasi server Anda ke Amazon. GameLift Untuk informasi lebih lanjut lihat, GetComputeAuthToken. Jenis: `char[]` Wajib: Ya

F StartMatchBackfillRequest

Informasi yang digunakan untuk membuat permintaan pengisian ulang perjodohan. Server game mengkomunikasikan informasi ini ke Amazon GameLift dalam satu StartMatchBackfill() panggilan.

Properti	Deskripsi
GameSessionArn	Pengidentifikasi sesi permainan yang unik. Operasi API `GetGameSessionId` mengembalikan pengenal dalam format ARN. Jenis: `char[]` 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: `char[]` 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: `TArray<FPlayer>` Wajib: Ya
TicketId	Pengenal unik untuk tiket permintaan pencocokan atau pencocokan isi ulang. Jika Anda tidak memberikan nilai, Amazon GameLift menghasilkannya. Gunakan pengidentifikasi ini

Properti	Deskripsi
	untuk melacak status tiket backfill match atau membatalkan permintaan jika diperlukan. Jenis: `char[]` 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 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 mengevalu asi latensi pemain, pemain harus melaporkan latensi untuk dicocokkan. Jenis: `TMap>FString, int32<` 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. Untuk informasi selengkapnya tentang atribut pemain, lihat AttributeValue.

Properti	Deskripsi
	Jenis: `TMap>FString, FAttribut eValue<` 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

F GameLiftDescribePlayerSessionsRequest

Objek yang menentukan sesi pemain mana yang akan diambil. Proses server menyediakan informasi ini dengan DescribePlayerSessions() panggilan ke Amazon GameLift.

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`FString`. `GameSessionID` Ini adalah string ID kustom atau Jenis: `std::string` Wajib: Tidak

Properti	Deskripsi
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 ID pemain. 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 dikembali kan. Jika Anda memberikan ID sesi pemain, parameter ini diabaikan. Jenis: `int` Wajib: Tidak

F StopMatchBackfillRequest

Informasi yang digunakan untuk membatalkan permintaan pengisian ulang perjodohan. Server game mengkomunikasikan informasi ini ke GameLift layanan Amazon 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

F AttributeValue

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 `FAttributeType` 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

F GameLiftGetFleetRoleCredentialsRequest

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.

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 kredensyal peran. Jenis: `FString` Wajib: Tidak

F GameLiftLongOutcome

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 “F GameLiftError” Wajib: Tidak

F GameLiftStringOutcome

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan. Jenis: `FString` Wajib: Tidak

Properti	Deskripsi
ResultWithOwnership	Hasil dari tindakan, dilemparkan sebagai rvalue, sehingga kode panggilan dapat mengambil kepemilikan objek. 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 “F GameLiftError” Wajib: Tidak

F GameLiftDescribePlayerSessionsOutcome

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan. Jenis: the section called “F GameLiftDescribePlayerSessionsResult” Wajib: Tidak
ResultWithOwnership	Hasil dari tindakan, dilemparkan sebagai rvalue, sehingga kode panggilan dapat mengambil kepemilikan objek.

Properti	Deskripsi
	Jenis: `FGameLiftDescribePlayerSessionsResult&&` 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 “F GameLiftError” Wajib: Tidak

F GameLiftDescribePlayerSessionsResult

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

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 “F GameLiftError” Wajib: Tidak

F 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 “F GameLiftError” Wajib: Tidak

F GameLiftPlayerSession

Properti	Deskripsi
CreationTime	Jenis: `long`

Properti	Deskripsi
	Wajib: Ya
FleetId	Jenis: `FString` Wajib: Ya
GameSessionId	Jenis: `FString` Wajib: Ya
IpAddress	Jenis: `FString` Wajib: Ya
PlayerData	Jenis: `FString` Wajib: Ya
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

F GameLiftGetComputeCertificateOutcome

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan. Jenis: the section called “F GameLiftGetComputeCertificateResult” Wajib: Tidak
ResultWithOwnership	Hasil dari tindakan, dilemparkan sebagai rvalue, sehingga kode panggilan dapat mengambil kepemilikan objek. 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 “F GameLiftError” Wajib: Tidak

F GameLiftGetComputeCertificateResult

Jalur ke sertifikat TLS pada komputasi Anda dan nama host komputasi.

Properti	Deskripsi
CertificatePath	Jenis: `FString` Wajib: Ya
ComputeName	Jenis: `FString` Wajib: Ya

F GameLiftGetFleetRoleCredentialsOutcome

Tipe data ini dihasilkan dari tindakan dan menghasilkan objek dengan properti berikut:

Properti	Deskripsi
Hasil	Hasil dari tindakan. Jenis: the section called “F GetFleetRoleCredentialsResult” 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.

Properti	Deskripsi
	Jenis: the section called “F GameLiftError” Wajib: Tidak

F GetFleetRoleCredentialsResult

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` 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`

Properti	Deskripsi
	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

F GameLiftError

Properti	Deskripsi
ErrorType	Jenis kesalahan. Tipe: A `GameLiftErrorType` enum. Wajib: Tidak
ErrorName	Nama kesalahannya. Jenis: `std::string` Wajib: Tidak

Properti	Deskripsi
ErrorMessage	Pesan kesalahan. Jenis: `std::string` Wajib: Tidak

Enum

Enum yang didefinisikan untuk SDK GameLift server Amazon (Unreal) didefinisikan sebagai berikut:

F AttributeType

- 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 ke Amazon gagal. GameLift
- NETWORK_NOT_INITIALIZED — Jaringan belum diinisialisasi.
- GAMESESSION_ID_NOT_SET — ID sesi permainan belum ditetapkan.
- BAD_REQUEST_EXCEPTION
- INTERNAL_SERVICE_EXCEPTION
- ALREADY_INITIALIZED — Server GameLift Amazon atau Klien telah diinisialisasi dengan Initialize ().
- FLEET_MISMATCH — Armada target tidak cocok dengan armada GameSession atau PlayerSession.
- GAMELIFT_CLIENT_NOT_INITIALIZED - Klien Amazon belum diinisialisasi. GameLift
- GAMELIFT_SERVER_NOT_INITIALIZED — Server Amazon belum diinisialisasi. GameLift

- GAME_SESSION_ENDED_FAILED — GameLift Amazon Server SDK tidak dapat menghubungi layanan untuk melaporkan sesi permainan berakhir.
- GAME_SESSION_NOT_READY — Sesi Game Server GameLift Amazon tidak diaktifkan.
- GAME_SESSION_READY_FAILED — GameLift Amazon Server SDK tidak dapat menghubungi layanan untuk melaporkan sesi permainan sudah siap.
- INITIALIZATION_MISMATCH — Metode klien dipanggil setelah Server: :Initialize (), atau sebaliknya.
- NOT_INITIALIZED — GameLift Server Amazon 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 - GameLift Amazon Server SDK tidak 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 - GameLift Amazon Server SDK tidak dapat menghubungi layanan untuk melaporkan proses siap.
- SDK_VERSION_DETECTION_FAILED — Deteksi versi SDK gagal.
- STX_CALL_FAILED — Panggilan ke komponen backend server XSTx telah gagal.
- STX_INITIALIZATION_FAILED - Komponen backend server XSTx gagal 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.

E 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.

E PlayerSessionStatus

- AKTIF
- SELESAI
- NOT_SET
- DIPESAN
- TIMEDOUT

Referensi SDK 3.x server Amazon GameLift Unreal Engine

Anda dapat menggunakan referensi SDK 3.x server Amazon GameLift Unreal Engine ini untuk membantu Anda mempersiapkan game multipemain untuk digunakan dengan Amazon. GameLift Untuk detail tentang proses integrasi, lihatTambahkan Amazon GameLift ke server game Anda.

Topik

- Referensi SDK GameLift server Amazon untuk Unreal Engine: Tindakan
- Referensi SDK GameLift server Amazon untuk Unreal Engine: Tipe data

Referensi SDK GameLift server Amazon untuk Unreal Engine: Tindakan

Referensi Amazon GameLift Server SDK ini dapat membantu Anda mempersiapkan proyek game Unreal Engine Anda untuk digunakan dengan Amazon. GameLift Untuk informasi detail tentang proses integrasi, lihat Tambahkan Amazon GameLift ke server game Anda.

API ini didefinisikan dalam `GameLiftServerSDK.h` dan `GameLiftServerSDKModels.h`.

Untuk menyiapkan plugin Unreal Engine dan melihat contoh kode Integrasikan Amazon GameLift ke dalam proyek Unreal Engine.

- Tindakan

- Tipe data

AcceptPlayerSession()

Memberitahu GameLift layanan Amazon bahwa pemain dengan ID sesi pemain tertentu telah terhubung ke proses server dan memerlukan validasi. Amazon GameLift memverifikasi bahwa ID sesi pemain valid—yaitu, ID pemain telah memesan slot pemain dalam sesi permainan. Setelah divalidasi, Amazon GameLift mengubah status slot pemain dari RESERVED menjadi AKTIF.

Sintaks

```
FGameLiftGenericOutcome AcceptPlayerSession(const FString& playerSessionId)
```

Parameter

playerSessionId

ID unik yang dikeluarkan oleh GameLift layanan Amazon sebagai respons terhadap panggilan ke tindakan AWS CreatePlayerSessionSDK Amazon GameLift API. Client game mereferensi ID ini saat menghubungkan ke proses server.

Jenis: String

Wajib: Ya

Nilai kembali

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

ActivateGameSession()

Memberitahu GameLift layanan Amazon bahwa proses server telah mengaktifkan sesi permainan dan sekarang siap untuk 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 kembali

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 F DescribePlayerSessionsRequest yang menjelaskan sesi pemain mana yang diambil.

Wajib: Ya

Nilai kembali

Jika berhasil, ini mengembalikan objek F DescribePlayerSessionsRequest yang berisi satu set objek sesi pemain yang sesuai dengan parameter permintaan. Objek sesi pemain memiliki struktur yang identik dengan tipe PlayerSessiondata AWS SDK Amazon GameLift 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 kembali

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-yang dikaitkan dengan armada dan instansinya. AWS Certificate Managermenghasilkan sertifikat ini ketika Anda membuat armada baru dengan konfigurasi sertifikat diatur 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 kembali

Jika berhasil, mengembalikan `GetInstanceCertificateOutcome` objek yang berisi lokasi file sertifikat TLS armada dan rantai sertifikat, yang disimpan pada instance. File sertifikat root, yang diekstrak dari rantai sertifikat, juga disimpan pada instance. Jika tidak berhasil, ini mengembalikan pesan kesalahan.

Untuk informasi selengkapnya tentang data sertifikat dan 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 kembali

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 SDK. Metode ini harus dipanggil pada peluncuran, sebelum inisialisasi GameLift terkait Amazon lainnya terjadi.

Sintaks

```
FGameLiftGenericOutcome InitSDK()
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai kembali

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

ProcessEnding()

Memberitahu GameLift layanan Amazon 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.

Sintaks

```
FGameLiftGenericOutcome ProcessEnding()
```

Parameter

Tindakan ini tidak memiliki parameter.

Nilai kembali

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

ProcessReady()

Memberitahu GameLift layanan Amazon bahwa proses server siap untuk menyelenggarakan sesi game. 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

F ProcessParameters

Sebuah objek F ProcessParameters yang mengomunikasikan informasi berikut tentang proses server:

- Nama metode callback, diimplementasikan dalam kode server game, yang dipanggil GameLift layanan Amazon untuk berkomunikasi dengan proses server.
- Nomor port yang didengarkan oleh proses server.
- Jalur ke file khusus sesi game apa pun yang Anda GameLift ingin Amazon tangkap dan simpan.

Wajib: Ya

Nilai kembali

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Contoh

Lihat kode sampel di Menggunakan Plugin Unreal Engine.

RemovePlayerSession()

Memberitahu GameLift layanan Amazon bahwa pemain dengan ID sesi pemain tertentu telah terputus dari proses server. Sebagai tanggapan, Amazon GameLift mengubah slot pemain menjadi tersedia, yang memungkinkannya ditugaskan ke pemain baru.

Sintaks

```
FGameLiftGenericOutcome RemovePlayerSession(const FString& playerSessionId)
```

Parameter

playerSessionId

ID unik yang dikeluarkan oleh GameLift layanan Amazon sebagai respons terhadap panggilan ke tindakan AWS CreatePlayerSessionSDK Amazon GameLift API. Client game mereferensi ID ini saat menghubungkan ke proses server.

Jenis: String

Wajib: Ya

Nilai kembali

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 denganFlexMatch. Lihat juga tindakan 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, GameLift layanan Amazon mengirimkan 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

F StartMatchBackfillRequest

Sebuah objek F StartMatchBackfillRequest yang mengkomunikasikan informasi berikut:

- ID tiket untuk ditetapkan ke permintaan backfill. Informasi ini bersifat opsional; jika tidak ada ID yang disediakan, Amazon GameLift akan membuat satu secara otomatis.
- Matchmaker untuk dikirimi 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 kembali

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 tindakan AWS SDK StopMatchmaking(). Pelajari lebih lanjut tentang fitur FlexMatch isi ulang.

Sintaks

```
FGameLiftGenericOutcome StopMatchBackfill (FStopMatchBackfillRequest
 &stopBackfillRequest);
```

Parameter

StopMatchBackfillRequest

Sebuah objek F StopMatchBackfillRequest yang mengidentifikasi tiket matchmaking untuk membatalkan:

- ID tiket yang ditetapkan ke permintaan backfill yang dibatalkan
- matchmaker yang dikirimi permintaan backfill
- sesi game yang terkait dengan permintaan backfill

Wajib: Ya

Nilai kembali

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

TerminateGameSession()

Metode ini tidak lagi digunakan dengan versi 4.0.1. Sebagai gantinya, proses server harus menelepon ProcessEnding() setelah sesi permainan berakhir.

Memberitahu GameLift layanan Amazon bahwa proses server telah mengakhiri sesi game 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 sinyal ke Amazon GameLift bahwa proses server segera tersedia untuk menjadi tuan rumah 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 kembali

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()`tindakan di Amazon GameLift Service API Reference).

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 kembali

Mengembalikan hasil generik yang terdiri dari keberhasilan atau kegagalan dalam sebuah pesan kesalahan.

Referensi SDK GameLift server Amazon untuk Unreal Engine: Tipe data

Referensi Amazon GameLift Server SDK ini dapat membantu Anda mempersiapkan proyek game Unreal Engine Anda untuk digunakan dengan Amazon. GameLift Untuk informasi detail tentang proses integrasi, lihat Tambahkan Amazon GameLift ke server game Anda.

API ini didefinisikan dalam `GameLiftServerSDK.h` dan `GameLiftServerSDKModels.h`.

Untuk menyiapkan plugin Unreal Engine dan melihat contoh kode Integrasikan Amazon GameLift ke dalam proyek Unreal Engine.

- Aksi
- Jenis Data

F DescribePlayerSessionsRequest

Jenis data ini digunakan untuk menentukan sesi pemain untuk diambil. Anda bisa menggunakannya sebagai berikut:

- Menyediakan PlayerSessionId untuk meminta sesi pemain tertentu.
- Menyediakan GameSessionId untuk meminta semua sesi pemain dalam sesi permainan yang ditentukan.
- Menyediakan PlayerId untuk meminta semua sesi pemain untuk pemain tertentu.

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 NextTokenuntuk 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. ID Pemain ditentukan oleh developer. Lihat Hasilkan ID pemain.

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

F ProcessParameters

Tipe data ini berisi kumpulan parameter yang dikirim ke GameLift layanan Amazon 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.

Tipe: Bilangan Bulat

Wajib: Ya

logParameters

Obyek dengan daftar jalur direktori untuk file log sesi game.

Jenis: TArray <FString>

Wajib: Tidak

onStartGameSesi

Nama fungsi callback yang dipanggil GameLift layanan Amazon untuk mengaktifkan sesi game baru. Amazon GameLift memanggil fungsi ini sebagai respons terhadap permintaan klien CreateGameSession. Fungsi callback mengambil GameSessionobjek (didefinisikan dalam Amazon GameLift Service API Reference).

Tipe: F OnStartGameSession

Wajib: Ya

onProcessTerminate

Nama fungsi callback yang dipanggil GameLift layanan Amazon untuk memaksa proses server dimatikan. Setelah memanggil fungsi ini, Amazon GameLift menunggu lima menit hingga proses server dimatikan dan merespons dengan ProcessEnding() panggilan sebelum mematikan proses server.

Tipe: F SimpleDelegate

Wajib: Tidak

onHealthCheck

Nama fungsi callback yang dipanggil GameLift layanan Amazon untuk meminta laporan status kesehatan dari proses server. Amazon GameLift memanggil fungsi ini setiap 60 detik. Setelah memanggil fungsi ini Amazon GameLift menunggu 60 detik untuk respons, dan jika tidak ada yang diterima. mencatat proses server sebagai tidak sehat.

Tipe: F OnHealthCheck

Wajib: Tidak

onUpdateGameSesi

Nama fungsi callback yang dipanggil GameLift layanan Amazon untuk meneruskan objek sesi game yang diperbarui ke proses server. Amazon GameLift memanggil fungsi ini ketika permintaan isi ulang pertandingan telah diproses untuk menyediakan data mak comblang yang diperbarui. Ini melewati GameSessionobjek, pembaruan status (`updateReason`), dan ID tiket isi ulang pertandingan.

Tipe: F OnUpdateGameSession

Wajib: Tidak

F StartMatchBackfillRequest

Jenis data ini digunakan untuk mengirim permintaan backfill matchmaking. Informasi tersebut dikomunikasikan ke GameLift layanan Amazon dalam StartMatchBackfill() panggilan.

Daftar Isi

GameSessionArn

Pengidentifikasi sesi game yang unik. Tindakan API GetGameSessionId() mengembalikan pengidentifikasi dalam format ARN.

Jenis: String

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: String

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 Panduan Referensi GameLift API Amazon untuk deskripsi format objek Player. Untuk menemukan atribut pemain, ID, dan tugas tim, lihat di objek sesi game, di properti data matchmaker. 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 disediakan di sini, Amazon GameLift akan menghasilkan satu dalam bentuk UUID. Gunakan pengidentifikasi ini untuk melacak status tiket backfill match atau membatalkan permintaan jika diperlukan.

Jenis: String

Wajib: Tidak

F StopMatchBackfillRequest

Jenis data ini digunakan untuk membatalkan permintaan backfill matchmaking. Informasi tersebut dikomunikasikan ke GameLift layanan Amazon dalam StopMatchBackfill() panggilan.

Daftar Isi

GameSessionArn

Pengidentifikasi sesi game unik yang terkait dengan permintaan yang dibatalkan.

Jenis: String

Wajib: Ya

MatchmakingConfigurationArn

Pengidentifikasi unik dari matchmaker sebagai tujuan pengiriman permintaan ini.

Jenis: String

Wajib: Ya

TicketId

Pengidentifikasi unik dari tiket backfill match yang akan dibatalkan.

Jenis: String

Wajib: Ya

Acara penempatan sesi game

Amazon GameLift mengeluarkan 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 secara real time dan atas dasar usaha 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 GameSessionPlacementdi Amazon GameLift API Reference.

Sintaksis peristiwa penempatan

Peristiwa direpresentasikan sebagai objek JSON. Struktur acara sesuai dengan pola CloudWatch Acara, 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

AWSAkun yang digunakan untuk mengelola AmazonGameLift.

Waktu

Stempel waktu peristiwa.

wilayah

Wilayah AWS di mana 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.

Sintaks detail:

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`.

Armada yang tidak mendukung TLS:`ec2-<unique identifier>.compute.amazonaws.com`.

startTime

Stempel waktu menunjukkan kapan permintaan ini ditempatkan dalam antrean.

endTime

Stempel waktu menunjukkan kapan permintaan ini terpenuhi.

gameSessionRegion

Wilayah AWS di mana sesi game sedang di-host. Informasi ini juga dalam gameSessionArn nilai.

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-123213123232412412 3123"
      }
    ]
  }
}
```


PlacementCancelled

Permintaan penempatan dibatalkan dengan panggilan ke layanan. GameLift [StopGameSessionPlacement]

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": "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 tidak dapat memenuhi permintaan sesi game. 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 antrean.

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"
  }
}
```

Menghasilkan perkiraan GameLift harga Amazon

DenganAWS Pricing Calculator, Anda dapat membuat perkiraan harga untuk Amazon GameLift. Anda tidak perlu Akun AWS atau pengetahuan mendalam AWS untuk menggunakan kalkulator.

AWS Pricing Calculatorkalkulator memandu Anda melalui keputusan yang memengaruhi biaya layanan untuk memberi Anda gambaran tentang berapa biaya Amazon GameLift untuk proyek game Anda. Jika Anda belum yakin bagaimana Anda berencana untuk menggunakan AmazonGameLift, maka gunakan nilai default untuk menghasilkan estimasi. Saat merencanakan penggunaan produksi, kalkulator dapat membantu Anda menguji potensi skenario dan menghasilkan perkiraan yang lebih akurat.

Anda dapat menggunakan AWS Pricing Calculator untuk menghasilkan perkiraan untuk opsi GameLift hosting Amazon berikut:

- Perkirakan GameLift hosting Amazon
- Perkirakan Amazon GameLift mandiri FlexMatch

Perkirakan GameLift hosting Amazon

Opsi ini memberikan perkiraan biaya untuk hosting game Anda di server yang GameLift dikelola Amazon, termasuk biaya untuk penggunaan instans server dan transfer data. FlexMatchperjodohan sudah termasuk dalam biaya untuk hosting yang GameLift dikelola Amazon.

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

GameLiftInstans Amazon

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

Untuk memulai, buka GameLiftthalaman Konfigurasi Amazon dariAWS Pricing Calculator. Tambahkan Deskripsi, pilih Wilayah, lalu pilih Perkirakan GameLift hosting Amazon (Instance + Transfer Data Out). Di bawah GameLiftinstans Amazon, 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 yang GameLift dibutuhkan Amazon untuk 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**.

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

  Ini adalah jumlah rata-rata pemain bersamaan per jam selama periode 24 jam. Kami menggunakan nilai ini untuk memperkirakan jumlah kapasitas hosting berkelanjutan yang GameLift perlu dipertahankan Amazon 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 per jam rata-rata 6.000 dan CCU puncak 10.000, maka masukkan nilai persen. **60**

- Sesi game per instance

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

- Pemain per sesi game

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

- Instance buffer siaga%

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

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

- Contoh spot%

  GameLiftArmada Amazon 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. Sebaiknya gunakan kombinasi untuk mengoptimalkan penghematan biaya dan ketersediaan. Untuk informasi tentang cara Amazon GameLift menggunakan Instans Spot, lihatInstans Sesuai Permintaan versus Instans Spot.

  Untuk bidang ini, masukkan persentase Instans Spot untuk dipelihara dalam armada. Kami merekomendasikan persentase Instans Spot antara 50 dan 85 persen. Nilai default adalah **`50`** persen.

  Misalnya, jika Anda menerapkan armada dengan 100 instans dan menentukan **`40`** persen, Amazon GameLift berfungsi untuk mempertahankan 60 Instans Sesuai Permintaan dan 40 Instans Spot.

- Tipe instans

  GameLiftArmada Amazon dapat menggunakan berbagai jenis instans Amazon EC2 yang bervariasi dalam kemampuan daya komputasi, memori, penyimpanan, dan jaringan. Saat Anda mengonfigurasi GameLift armada Amazon, pilih jenis instans yang paling sesuai dengan kebutuhan game Anda. Untuk informasi tentang memilih jenis instans dengan AmazonGameLift, lihatMemilih AmazonGameLiftmenghitung sumber daya.

  Jika Anda mengetahui jenis instans yang Anda gunakan atau rencanakan untuk digunakan di GameLift armada Amazon Anda, 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 untuk 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 GameLiftthalaman Konfigurasi AmazonAWS Pricing Calculator, perluas Transfer data keluar (DTO), lalu lengkapi bidang berikut:

- Jenis perkiraan DTO

  Anda dapat memilih untuk memperkirakan DTO dengan 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), maka 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 estimasi GameLift harga Amazon, pilih Tambahkan ke perkiraan saya. Untuk informasi selengkapnya tentang membuat dan mengelola estimasiAWS Pricing Calculator, lihat Membuat estimasi, mengonfigurasi layanan, dan menambahkan lebih banyak layanan di Panduan AWS Pricing Calculator Pengguna.

Perkirakan Amazon GameLift mandiri FlexMatch

Opsi ini memberikan perkiraan biaya untuk menggunakan FlexMatch perjodohan sebagai layanan mandiri saat menghosting game Anda dengan solusi server game lain. Ini termasuk hosting yang

GameLift dikelola sendiri Amazon dengan FleetiQ dan hosting lokal,peer-to-peer, atau tipe data primitif komputasi cloud. FlexMatchBiaya mandiri didasarkan pada daya komputasi yang digunakan.

Jika Anda memiliki atau berencana untuk memiliki lebih dari satu mak comblang di AWS Wilayah yang berbeda, buat perkiraan untuk setiap Wilayah.

> **Note**
>
> Amazon GameLift FlexMatch tersedia di Wilayah berikut: AS Timur (Virginia), AS Barat (Oregon), Asia Pasifik (Seoul), Asia Pasifik (Sydney), Asia Pasifik (Tokyo), Eropa (Frankfurt), Eropa (Irlandia).

Untuk memulai, buka GameLiftthalaman Konfigurasi Amazon dariAWS Pricing Calculator. Tambahkan Deskripsi, pilih Wilayah, lalu pilih Perkirakan Amazon GameLift Standalone FlexMatch. Di bawah Amazon GameLift FlexMatch, lengkapi bidang-bidang berikut:

- Puncak pemain bersamaan (puncak CCU)

  Ini adalah jumlah maksimum pemain yang dapat terhubung ke server game Anda pada saat yang sama dan meminta perjodohan. Masukkan jumlah puncak harian pemain yang Anda harapkan untuk mencocokkan ke sesi permainan di Wilayah pilihan Anda.

  Misalnya, jika Anda ingin mencocokkan sebanyak 1.000 pemain sekaligus, pertahankan nilai default**`1000`**.
- Rata-rata CCU per jam sebagai persentase puncak CCU harian

  Ini adalah jumlah rata-rata pemain bersamaan per jam selama periode 24 jam. Nilai ini membantu memperkirakan volume permintaan perjodohan 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 per jam rata-rata 6.000 dan CCU puncak 10.000, maka masukkan nilai persen. **`60`**
- Jumlah pemain per pertandingan

  Ini adalah jumlah rata-rata pemain yang cocok dengan sesi permainan, sebagaimana didefinisikan dalam desain game Anda. Jika Anda memiliki mode permainan dengan jumlah pemain yang

berbeda, perkirakan jumlah pemain rata-rata per sesi game di seluruh game Anda. Nilai default-nya adalah **8**.

- Durasi permainan (dalam menit)

  Ini adalah durasi rata-rata waktu pemain tetap dalam sesi permainan dari awal hingga akhir. Nilai ini membantu menentukan seberapa sering pemain mungkin memerlukan pertandingan baru. Masukkan durasi permainan rata-rata dalam hitungan menit untuk pemain Anda. Nilai default-nya adalah **1**.

- Kompleksitas aturan perjodohan

  Kompleksitas aturan perjodohan mengacu pada jumlah dan jenis aturan yang Anda gunakan untuk mencocokkan pemain. Tingkat kompleksitas set aturan Anda membantu menentukan jumlah daya komputasi yang diperlukan untuk setiap pertandingan.

  - Kompleksitas lebih rendah - Pilih opsi ini jika set aturan perjodohan Anda mencakup beberapa aturan, menggunakan jenis aturan yang lebih sederhana (seperti aturan perbandingan), dan memiliki aturan yang membentuk kecocokan yang berhasil dengan upaya yang lebih sedikit.
  - Kompleksitas yang lebih tinggi — Pilih opsi ini jika kumpulan aturan perjodohan Anda mencakup beberapa aturan, menggunakan jenis aturan yang lebih kompleks (seperti aturan jarak atau latensi), dan memiliki aturan ketat yang mengakibatkan lebih banyak kegagalan dan memerlukan lebih banyak upaya pencocokan.

  Untuk informasi selengkapnya tentang kompleksitas aturan dan harga, lihat Amazon GameLift FlexMatch di halaman GameLift Harga Amazon.

Setelah selesai mengonfigurasi estimasi GameLift FlexMatch harga Amazon, pilih Tambahkan ke perkiraan saya. Untuk informasi selengkapnya tentang membuat dan mengelola estimasiAWS Pricing Calculator, lihat Membuat estimasi, mengonfigurasi layanan, dan menambahkan lebih banyak layanan di Panduan AWS Pricing Calculator Pengguna.

Kuota dan Wilayah yang didukung

Untuk kuota GameLift layanan AWS Amazon, lihat GameLiftkuota Amazon.

Untuk informasi tentang meminta peningkatan kuota untuk AWS sumber daya, lihat AWS lquota layanan.

Untuk daftar Amazon yang Wilayah AWS mendukungGameLift, lihat GameLiftWilayah Amazon.

Catatan GameLift rilis Amazon

Catatan GameLift rilis Amazon memberikan detail tentang fitur, pembaruan, dan perbaikan baru yang terkait dengan layanan.

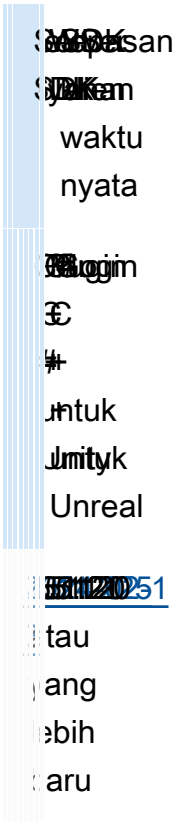
Versi SDK

Tabel berikut mencantumkan semua GameLift rilis Amazon dengan informasi versi SDK. Tidak ada persyaratan untuk menggunakan SDK yang sebanding untuk server game dan integrasi klien Anda. Namun, versi sebelumnya dari satu SDK mungkin tidak sepenuhnya mendukung fitur terbaru di SDK lainnya.

Untuk informasi selengkapnya tentang Amazon GameLift SDK, lihatDukungan pengembangan dengan Amazon GameLift.

Untuk mendapatkan GameLift SDK Amazon terbaru, lihat situs unduhan Amazon GameLift SDK.

Versi saat ini



Versi sebelumnya

Pelepasan layanan	AWS SDK	Server SDK				SDK klien waktu nyata
		Plugin C # untuk Unity	C++	Plugin C ++ untuk Unreal	Go	
2023-12-14	1.11.225 atau yang lebih baru	5.1.0	5.1.1	5.1.0	5.0.0	1.2.0
2023-11-02	1.11.193 atau yang lebih baru	5.1.0	5.1.1	5.1.0	5.0.0	1.2.0
2023-09-28	1.11.144 atau yang lebih baru	5.1.0	5.1.1	5.1.0	5.0.0	1.2.0
2023-08-17	1.11.144 atau yang lebih baru	5.1.0	5.1.1	5.1.0	5.0.0	1.2.0
2023-07-27	1.11.111 atau yang lebih baru	5.1.0	5.1.0	5.0.2	5.0.0	1.2.0
2023-06-29	1.11.111 atau yang lebih baru		5.0.4	5.0.2	5.0.0	1.2.0

Pelepasan layanan	AWS SDK	Server SDK				SDK klien waktu nyata
		Plugin C # untuk Unity	C++	Plugin C ++ untuk Unreal	Go	
2023-06-15	1.11.87 atau lebih baru		5.0.4	5.0.2	5.0.0	1.2.0
2023-05-25	1.11.87 atau lebih baru		5.0.3	5.0.2	5.0.0	1.2.0
2023-04-20	1.11.63 atau yang lebih baru				5.0.0	1.2.0
2023-04-13	1.10.21 atau yang lebih baru				5.0.0	1.2.0
2023-02-09	1.10.21 atau yang lebih baru			3.4.0	5.0.0	1.2.0
2023-01-31	1.10.21 atau yang lebih baru			3.4.0	5.0.0	1.2.0
2022-12-01	1.10.21 atau yang lebih baru			3.4.0		1.2.0
2022-08-25	1.9.333 atau yang lebih baru		3.4.2	3.4.0		1.2.0

Pelepasan layanan	AWS SDK	Server SDK				SDK klien waktu nyata
		Plugin C # untuk Unity	C++	Plugin C ++ untuk Unreal	Go	
2021-10-28	1.9.133 atau yang lebih baru		3.4.2	3.4.0		1.2.0
2021-06-03	1.8.168 atau yang lebih baru		3.4.2	3.4.0		1.2.0
2021-03-23	1.8.168 atau yang lebih baru		3.4.1	3.3.3		1.1.0
2021-03-16	1.8.163 atau yang lebih baru		3.4.1	3.3.3		1.1.0
2021-02-09	1.8.139 atau yang lebih baru		3.4.1	3.3.3		1.1.0
2020-12-22	1.8.95 atau yang lebih baru		3.4.1	3.3.3		1.1.0
2020-11-24	1.8.95 atau yang lebih baru		3.4.1	3.3.2		1.1.0
2020-11-11	1.8.36 atau yang lebih baru		3.4.1	3.3.2		1.1.0

Pelepasan layanan	AWS SDK	Server SDK				SDK klien waktu nyata
		Plugin C # untuk Unity	C++	Plugin C ++ untuk Unreal	Go	
2020-09-17	1.8.36 atau yang lebih baru		3.4.1	3.3.2		1.1.0
2020-08-27	1.7.310 atau yang lebih baru		3.4.0	3.3.1		1.1.0
2020-04-16	1.7.310 atau yang lebih baru		3.4.0	3.3.1		1.1.0
2020-04-02	1.7.310 atau yang lebih baru		3.4.0			1.1.0
2019-12-19	1.7.249 atau yang lebih baru		3.4.0			1.1.0
2019-11-14	1.7.210 atau yang lebih baru		3.4.0			1.1.0
2019-10-24	1.7.210 atau yang lebih baru		3.4.0			1.1.0
2019-09-03	1.7.175 atau yang lebih baru		3.4.0			1.1.0

Pelepasan layanan	AWS SDK	Server SDK				SDK klien waktu nyata
		Plugin C # untuk Unity	C++	Plugin C ++ untuk Unreal	Go	
2019-07-09	1.7.140 atau yang lebih baru		3.3.0			1.0.0
2019-04-25	1.7.91 atau yang lebih baru		3.3.0			1.0.0
2019-03-07	1.7.65 atau yang lebih baru		3.3.0			
2019-02-07	1.7.45 atau yang lebih baru		3.3.0			
2018-12-14	1.6.20 atau yang lebih baru		3.3.0			
2018-09-27	1.6.20 atau yang lebih baru		3.2.1			
2018-06-14	1.4.47 atau yang lebih baru		3.2.1			
2018-05-10	1.4.47 atau yang lebih baru		3.2.1			

Pelepasan layanan	AWS SDK	Server SDK				SDK klien waktu nyata
		Plugin C # untuk Unity	C++	Plugin C ++ untuk Unreal	Go	
2018-02-15	1.3.58 atau yang lebih baru		3.2.1			
2018-02-08	1.3.52 atau yang lebih baru		3.2.0			
2017-09-01	1.1.43 atau yang lebih baru		3.1.7			
2017-08-16	1.1.31 atau yang lebih baru		3.1.7			
2017-05-16	1.0.122 atau yang lebih baru		3.1.5			
2017-04-11	1.0.103 atau yang lebih baru		3.1.5			
2017-02-21	1.0.72 atau yang lebih baru		3.1.5			
2016-11-18	1.0.31 atau yang lebih baru		3.1.0			

Pelepasan layanan	AWS SDK	Server SDK				SDK klien waktu nyata
		Plugin C # untuk Unity	C++	Plugin C ++ untuk Unreal	Go	
2016-10-13	1.0.17 atau yang lebih baru		3.1.0			
2016-09-01	0.14.9 atau yang lebih baru		3.1.0			
2016-08-04	0.12.16 atau yang lebih baru		3.0.7			

Catatan rilis

Catatan rilis berikut dalam urutan kronologis, dengan perbaruan terbaru terdaftar terlebih dahulu. Amazon pertama kali GameLift dirilis pada tahun 2016. Untuk catatan rilis pada tanggal lebih awal dari yang tercantum di sini, lihat link tanggal rilis di Versi SDK.

24 April 2024: Amazon GameLift meluncurkan armada kontainer

Amazon GameLift sekarang menawarkan pratinjau armada kontainer, yang memberi Anda peningkatan portabilitas, skalabilitas, toleransi kesalahan, dan kelincahan.

Dalam armada kontainer, instans Amazon EC2 menampung satu atau beberapa kontainer Anda. Wadah ini termasuk server game Anda bersama dengan apa pun yang diperlukan, termasuk dependensi dan konfigurasi. Contoh dependensi termasuk SDK dan paket perangkat lunak. Setelah Anda mengunggah kontainer Anda ke Amazon Elastic Container Registry pribadi Anda, Amazon GameLift mengisi 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 perjodohan. 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 GameLift wadah Amazon, Panduan GameLift Pengembang Amazon
- CreateContainerGroupDefinition, Referensi GameLift API Amazon

13 Februari 2024: Amazon GameLift meluncurkan peningkatan pada SDK, dan menyederhanakan pemasangan plugin Amazon GameLift 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 di armada EC2 yang GameLift dikelola Amazon membaca parameter server langsung dari variabel lingkungan. Server game di GameLift Anywhere armada Amazon harus menelepon `InitSDK()` dengan parameter server.

Diperbarui versi plugin:

- GameLift Plugin Amazon untuk Unreal Engine, versi 1.1.0
- GameLift Plugin Amazon 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:

- [GameLift Plugin Amazon 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 GameLift Server SDK.

Pelajari lebih lanjut:

- Mengintegrasikan game dengan GameLift plugin Amazon untuk Unreal Engine, Panduan Pengembang Amazon GameLift
- GameLift Plugin Amazon dan unduhan SDK

14 Desember 2023: Amazon GameLift 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 ke klien game”, Panduan GameLift Pengembang Amazon
- GameProperty, Referensi GameLift API Amazon
- UpdateGameSession, Referensi GameLift API Amazon
- SearchGameSessions, Referensi GameLift API Amazon

21 November 2023: Amazon GameLift meluncurkan dukungan untuk Infrastruktur sebagai alat Kode seperti Terraform dan Pulumi yang didukung oleh AWS Cloud Control API

Sekarang Anda dapat mengelola seluruh tumpukan GameLift sumber daya Amazon 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 GameLift sumber daya Amazon dengan menggunakan AWS Cloud Control API. Anda dapat terus bekerja dengan sumber daya menggunakan GameLift API Amazon atau AWS CloudFormation templat untuk Amazon GameLift.

Untuk detail tentang GameLift sumber daya Amazon yang tersedia melalui IAc, lihat referensi jenis GameLift sumber daya Amazon referensi jenis GameLift sumber daya Amazon.

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 API standar 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 memperbarui plugin mandiri untuk Unity

Versi SDK yang diperbarui: GameLift Plugin Amazon untuk Unity, versi 2.0.0

GameLift Plugin Amazon untuk Unity menyediakan alat dan alur kerja yang merampingkan langkah-langkah untuk membuat game Unity Anda aktif dan berjalan untuk hosting cloud dengan Amazon. GameLift Amazon GameLift 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 GameLift fitur Amazon terbaru, termasuk server SDK versi 5.x dan dukungan untuk pengujian lokal dengan Amazon GameLift 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 menggunakan workstation lokal Anda sebagai host. Alur kerja ini membantu Anda menyiapkan GameLift Anywhere armada Amazon untuk mesin lokal Anda, meluncurkan instance server dan klien game Anda, meminta sesi game melalui Amazon GameLift, dan bergabung dengan game.

  - Terapkan solusi cloud hosting untuk server game terintegrasi Anda dengan EC2 yang GameLift dikelola Amazon dan sumber daya pendukung AWS . 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 GameLift integrasi dan penyebaran Amazon, termasuk:
  - Setiap solusi hosting mencakup AWS sumber daya pendukung, termasuk kumpulan pengguna Amazon Cognito yang menyediakan ID pemain unik dan validasi pemain. Solusinya juga mencakup bucket Amazon S3 untuk penyimpanan, notifikasi acara Amazon SNS, fungsi AWS Lambda , dan sumber daya lainnya.
  - Untuk Anywhere alur kerja, plugin mengotomatiskan pengaturan parameter server yang diperlukan.
  - Untuk alur kerja Amazon EC2, setiap solusi penerapan menyediakan layanan backend klien bawaan menggunakan fungsi Lambda. Layanan backend berada di antara klien game dan GameLift layanan Amazon dan mengelola semua panggilan langsung ke layanan Amazon. GameLift
- 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 Anywhere dan armada Amazon EC2, AWS CloudFormation menggunakan templat untuk mendeskripsikan dan menyebarkan sumber daya untuk solusi AWS game Anda. Template ini termasuk dalam unduhan GameLift plugin Amazon. Anda dapat menggunakannya apa adanya atau menyesuaikannya untuk game Anda.

Pelajari lebih lanjut:

- GameLift Plugin Amazon untuk panduan Unity untuk server SDK 5.x, Panduan GameLift Pengembang Amazon
- Unduh plugin dari GitHub

- Tentang Amazon GameLift hosting
- GameLift Forum Amazon

2 November 2023: Amazon GameLift menambahkan dukungan untuk kredensi bersama

Versi SDK yang diperbarui: AWS SDK 1.11.193

Fitur GameLift kredensi bersama Amazon yang baru memungkinkan aplikasi yang digunakan pada armada EC2 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 kredensional menggunakan aksi SDK 5.x server.) `GetFleetRoleCredentials()`

Misalnya, jika Anda ingin menerapkan build server game dengan CloudWatch agen Amazon untuk mengumpulkan metrik instans EC2 dan data lainnya, agen tersebut 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 kredenal bersama diaktifkan. Saat Amazon GameLift menerapkan build server game Anda ke setiap instans EC2, Amazon akan menghasilkan file kredensyal bersama dan menyimpannya di instans. Semua aplikasi pada instance dapat menggunakan kredensi bersama. Amazon GameLift secara otomatis menyegarkan kredensil sementara sepanjang masa pakai instans.

Anda dapat mengaktifkan kredensi bersama saat membuat armada EC2 terkelola menggunakan metode berikut:

- Dalam alur kerja pembuatan armada GameLift konsol Amazon.
- Saat memanggil operasi API GameLift layanan Amazon `CreateFleet` menggunakan parameter baru`InstanceRoleCredentialsProvider`.
- 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 GameLift Pengembang Amazon

- CreateFleet, InstanceRoleCredentialsProvider, Referensi GameLift API Amazon
- Siapkan peran layanan IAM, Panduan GameLift Pengembang Amazon

28 September 2023: Amazon GameLift merilis plugin mandiri baru untuk Unreal Engine

Versi SDK yang diperbarui: GameLift Plugin Amazon untuk Unreal Engine versi 1.0.0

GameLift Plugin Amazon untuk Unreal Engine menyediakan alat dan alur kerja yang merampingkan langkah Anda untuk memulai dan menjalankan game dengan Amazon GameLift untuk hosting cloud. Amazon GameLift adalah 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 menggunakan workstation lokal Anda sebagai host. Alur kerja ini membantu Anda menyiapkan GameLift Anywhere armada Amazon untuk mesin lokal Anda, meluncurkan instance server dan klien game Anda, meminta sesi game melalui Amazon GameLift, dan mendapatkan informasi koneksi untuk sesi game baru.
  - Terapkan solusi hosting cloud Amazon EC2 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 GameLift sumber daya Amazon 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 GameLift integrasi dan penyebaran Amazon, termasuk:
  - Setiap solusi hosting mencakup AWS sumber daya pendukung, termasuk kumpulan pengguna Amazon Cognito dasar yang menyediakan ID pemutar unik, bucket Amazon S3 untuk penyimpanan, pemberitahuan acara Amazon SNS, dan fungsi. AWS Lambda
  - Untuk Anywhere alur kerja, plugin mengotomatiskan pengaturan parameter server yang diperlukan menggunakan argumen baris perintah.

  - Untuk alur kerja Amazon EC2, setiap solusi penerapan menyediakan layanan backend klien bawaan menggunakan fungsi Lambda. Layanan backend menerima permintaan dari klien game dan meneruskannya ke layanan Amazon GameLift .
- 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 armada for Anywhere dan Amazon EC2, AWS CloudFormation menggunakan templat untuk menjelaskan solusinya. Plugin menggunakan template ini saat menyebarkan GameLift sumber daya Amazon untuk game Anda. Template ini termasuk dalam unduhan GameLift plugin Amazon dan dapat diedit. Anda dapat menggunakannya apa adanya atau memodifikasinya untuk permainan Anda.

Pelajari lebih lanjut:

- Mengintegrasikan game dengan GameLift plugin Amazon untuk Unreal Engine, Panduan GameLift Pengembang Amazon
- Unduh plugin dari GitHub
- Tentang Amazon GameLift hosting
- GameLift Forum Amazon

17 Agustus 2023: Amazon GameLift menawarkan hosting server game dengan prosesor AWS Graviton

Versi SDK yang diperbarui: AWS SDK 1.11.144

Dengan Amazon, GameLift Anda sekarang dapat meng-host game Anda di cloud menggunakan instans EC2 dengan prosesor AWS Graviton. Dirancang oleh AWS prosesor berbasis ARM64, instans Graviton memberikan kinerja harga terbaik untuk beban kerja cloud menggunakan 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, Anda sekarang dapat memilih dari instance baru ini di 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 instans EC2 berbasis 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 GameLift server Amazon 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 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 GameLift server Amazon memberikan pembaruan untuk C ++, C #, dan plugin Unreal, dan plugin baru untuk digunakan dengan mesin game Unity. Pengembang game mengintegrasikan SDK GameLift server Amazon ke server game yang mereka gunakan untuk hosting di Amazon. GameLift

Versi SDK server terbaru berisi pembaruan berikut, yang mencakup sejumlah permintaan pelanggan:

- Unduh paket SDK khusus bahasa — Situs GameLift unduhan Amazon 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 di armada EC2 yang GameLift dikelola Amazon membaca parameter server langsung dari variabel lingkungan. Server game di GameLift Anywhere armada Amazon 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.
    - Referensi SDK 5.x GameLift server Amazon untuk C # dan Unity
    - Referensi SDK 5.x GameLift server Amazon untuk C++
    - Referensi SDK GameLift 5.x server Amazon Unreal Engine
  - Versi baru dari panduan integrasi SDK 5 server untuk plugin Unity dan Unreal
    - Integrasikan Amazon GameLift ke dalam proyek Unity
    - Integrasikan Amazon GameLift ke dalam proyek Unreal Engine
- Pembaruan dokumentasi tambahan
  - Dokumentasi yang direvisi untuk operasi API GameLift layanan Amazon GetComputeAccessdan GetInstanceAccessuntuk memperjelas prosedur akses jarak jauh berdasarkan versi SDK GameLift server Amazon yang digunakan.
  - Deskripsi yang direvisi GameSessionPlacementuntuk mendokumentasikan bagaimana informasi sesi game bersifat sementara ketika penempatan berada dalam status “tertunda”.

13 Juli 2023: Amazon GameLift menambahkan metrik perangkat keras armada

Sekarang Anda dapat melacak metrik kinerja perangkat keras untuk armada EC2 yang GameLift dikelola Amazon. Metrik mencakup metrik instans EC2 untuk pemanfaatan CPU, volume lalu lintas jaringan, dan aktivitas baca/tulis disk. Untuk Amazon GameLift, metrik ini menjelaskan semua instans 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 GameLift konsol Amazon dalam detail armada.

Pelajari selengkapnya:

- Pantau Amazon GameLift dengan Amazon CloudWatch(Metrik untuk armada), Panduan Pengembang Amazon GameLift

29 Juni 2023: Amazon GameLift meluncurkan dukungan untuk Amazon Linux 2023

Versi SDK yang diperbarui: AWS SDK 1.11.111

GameLift Pelanggan Amazon sekarang dapat menggunakan sistem operasi Amazon Linux 2023 untuk meng-host server game mereka. AL2023 menawarkan beberapa peningkatan dibandingkan AL2 termasuk keamanan. Sistem operasi ini tersedia di semua Wilayah AWS kecuali Wilayah China.

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 2025.

Pelajari selengkapnya:

- FAQ Server Amazon GameLift Linux
- Membandingkan Amazon Linux 2 dan Amazon Linux 2023
- Tautan Referensi GameLift API Amazon:
  - AWS Tindakan SDK `CreateBuild`
  - Perintah CLI `upload-build`
  - Perintah CLI `create-build`

25 Mei 2023: Amazon GameLift 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 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, Amazon GameLift FleetiQ hanya melihat instance aktif saat mencari dan mengklaim server game yang tersedia.

Pelajari lebih lanjut:

- ClaimGameServerdi Referensi GameLift API Amazon
- Cara kerja FleetiQ di Panduan Pengembang Amazon GameLift FleetiQ

16 Mei 2023: Amazon GameLift mendukung penandaan alokasi biaya untuk armada

GameLift Pelanggan Amazon sekarang dapat menggunakan tag alokasi AWS Billing biaya untuk mengatur biaya hosting game mereka. Anda dapat menetapkan tag alokasi biaya ke sumber daya armada Amazon GameLift EC2 individual untuk melacak bagaimana armada Anda berkontribusi terhadap biaya hosting secara keseluruhan.

Pelajari lebih lanjut:

- Kelola biaya hosting game Anda
- Menggunakan tag alokasi AWS biaya, AWS Billing Panduan Pengguna

20 April 2023: Amazon GameLift meluncurkan dukungan untuk Windows Server 2016

Versi SDK yang diperbarui: AWS SDK 1.11.63

GameLift Pelanggan Amazon 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 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 GameLift konsol Amazon, 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:

- Tautan Referensi GameLift API Amazon:
  - Perintah CLI `upload-build`
  - Perintah CLI `create-build`
  - AWS Tindakan SDK `CreateBuild`
- Amazon GameLift FAQ untuk Windows 2012

13 April 2023: Amazon GameLift meluncurkan server SDK 5.x untuk Unreal

Versi SDK yang diperbarui: Server SDK 5.0.0 untuk Unreal

Versi terbaru dari plugin GameLift ringan Amazon untuk Unreal Engine sekarang didasarkan pada GameLift server Amazon SDK 5.x. Untuk mulai mengintegrasikan lingkungan Unreal Engine Anda dengan Amazon, GameLift lihat tautan berikut.

Pelajari lebih lanjut:

- Integrasikan Amazon GameLift ke dalam proyek Unreal Engine

- Tambahkan Amazon GameLift ke server game Anda
- Referensi SDK 5.x GameLift server Amazon untuk C++

14 Maret 2023: Amazon GameLift meluncurkan pengalaman konsol baru

GameLift Konsol Amazon baru mencakup peningkatan ini:

- Peningkatan navigasi - Panel navigasi baru memfasilitasi navigasi antara GameLift sumber daya Amazon.
- Halaman GameLift landing Amazon - Halaman arahan baru menyediakan tautan ke dokumentasi yang bermanfaat, menampilkan ikhtisar Amazon tingkat tinggi GameLift, dan memberikan dukungan melalui tautan ke dokumentasi, pertanyaan umum, dan AWS re:Post.
- CloudWatch Metrik Amazon yang disempurnakan — GameLift Metrik Amazon sekarang tersedia di GameLift konsol Amazon dan dasbor Anda CloudWatch. Pembaruan ini juga mencakup metrik baru untuk kinerja, pemanfaatan, dan sesi pemain.

Pelajari lebih lanjut:

- Melihat data game Anda di konsol
- Mengelola sumber daya GameLift hosting Amazon
- Membangun FlexMatch mak comblang

14 Februari 2023: Amazon GameLift 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
- FlexMatchacara perjodohan
- Enkripsi saat istirahat

9 Februari 2023: SDK GameLift server Amazon 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 GameLift server Amazon SDK 5.0.0 dengan.NET 4.6. Unity tidak mendukung .NET 6.

Pelajari lebih lanjut:

- Unduh versi terbaru SDK GameLift server Amazon di Amazon GameLift untuk memulai
- Referensi SDK 5.x GameLift server Amazon untuk C # dan Unity

31 Januari 2023: SDK GameLift server Amazon mendukung bahasa Go

Versi SDK yang diperbarui: Server SDK 5.0.0 untuk Go

Pelajari lebih lanjut:

- Unduh versi terbaru SDK GameLift server Amazon di Amazon GameLift untuk memulai
- Referensi SDK GameLift server Amazon untuk Go

1 Desember 2022: Amazon GameLift meluncurkan Amazon GameLift Anywhere dan Amazon GameLift Server SDK 5.0

Versi SDK yang diperbarui: AWS SDK 1.10.21, Server SDK 5.0.0 untuk C ++ dan C #

Amazon GameLift Anywhere menggunakan sumber daya server game Anda untuk meng-host server GameLift game Amazon. Anda dapat menggunakan Amazon GameLift Anywhere untuk mengintegrasikan sumber daya komputasi Anda sendiri dengan komputasi EC2 GameLift terkelola Amazon untuk mendistribusikan server game Anda di beberapa jenis komputasi. Anda juga dapat menggunakan Amazon GameLift Anywhere untuk menguji server game secara berulang tanpa mengunggah build ke Amazon GameLift untuk setiap iterasi.

Sorotan:

- GameLift AnywhereArmada Amazon baru dan jenis komputasi
- Amazon GameLift Anywhere menghitung pendaftaran sumber daya
- Peningkatan siklus iterasi pengujian

Amazon GameLift Server SDK 5.0.0 memperkenalkan peningkatan pada SDK server yang ada dan jenis sumber daya baru, komputasi. Server SDK 5.0.0 mendukung Amazon GameLift Anywhere dan penggunaan sumber daya komputasi Anda sendiri untuk hosting server game.

Pelajari lebih lanjut:

- Referensi SDK GameLift server Amazon
- Lokasi armada
- Memilih AmazonGameLiftmenghitung sumber daya
- Membuat GameLift Anywhere armada Amazon

25 Agustus 2022: Amazon GameLift meluncurkan dukungan untuk Local Zones

Versi SDK yang diperbarui: AWS SDK 1.9.333

Amazon sekarang GameLift tersedia di delapan Local Zones di Amerika Serikat, sehingga Anda dapat menyebarkan armada Anda lebih dekat ke pemain. Anda dapat menggunakan semua GameLift fitur Amazon yang dikelola 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:

- Zona Lokal
- Lokasi armada
- Buat armada GameLift terkelola Amazon

28 Juni 2022: Amazon GameLift meluncurkan pengalaman konsol keikutsertaan baru

GameLift Konsol Amazon baru mencakup peningkatan ini:

- Peningkatan navigasi - Panel navigasi baru memfasilitasi navigasi antara GameLift sumber daya Amazon.

- Halaman GameLift landing Amazon - Halaman arahan baru menyediakan tautan ke dokumentasi yang bermanfaat, menampilkan ikhtisar Amazon tingkat tinggi GameLift, dan memberikan dukungan melalui tautan ke dokumentasi, pertanyaan umum, dan AWS re:Post.
- CloudWatch Metrik Amazon yang disempurnakan — GameLift Metrik Amazon sekarang tersedia di GameLift konsol Amazon dan dasbor Anda CloudWatch. Pembaruan ini juga mencakup metrik baru untuk kinerja, pemanfaatan, dan sesi pemain.

Pelajari lebih lanjut:

- Melihat data game Anda di konsol
- Mengelola sumber daya GameLift hosting Amazon
- Membangun FlexMatch mak comblang

15 Februari 2022: FlexMatch menambahkan aturan majemuk dan perbaikan tambahan

FlexMatch pengguna sekarang memiliki akses ke fitur-fitur berikut:

- Aturan majemuk — Menambahkan dukungan untuk aturan perjodohan gabungan untuk pertandingan 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`, dan`xor`.
- Pemilihan tim yang fleksibel — Ekspresi properti perjodohan 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:

- Panduan GameLift FlexMatch pengembang Amazon:
  - FlexMatch jenis aturan
  - FlexMatch ekspresi properti
- AttributeValue: SL

28 Oktober 2021: Amazon GameLift menambahkan dukungan untuk armada Multi-wilayah di Wilayah Asia Pasifik (Osaka); Amazon FleetiQ menambahkan dukungan untuk prosesor GameLift Graviton2 AWS

Versi SDK yang diperbarui: AWS SDK 1.9.133

Amazon sekarang GameLift 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 sekarang GameLift tersedia di Wilayah Asia Pasifik (Osaka).
- Grup server game Amazon GameLift FleetiQ sekarang dapat dikonfigurasi untuk mengelola keluarga instans Graviton2 c6g, m6g, dan r6g.

Pelajari lebih lanjut:

- Armada GameLift Multi-wilayah Amazon
- CreateGameServerGroup
- AWS prosesor graviton

20 September 2021: Amazon GameLift merilis plugin untuk Unity

GameLift Plugin Amazon untuk Unity versi 1.0.0 berisi pustaka dan UI asli yang memudahkan akses GameLift sumber daya Amazon dan mengintegrasikan Amazon GameLift ke dalam game Unity Anda. Anda dapat menggunakan GameLift plugin Amazon untuk Unity untuk mengakses Amazon GameLift API dan menerapkan AWS CloudFormation template untuk skenario game umum. Plugin ini juga menyertakan contoh permainan yang bekerja dengan skenario sampel. Anda dapat menggunakan Amazon GameLift Local untuk melihat pesan yang diteruskan antara klien game dan server game untuk mempelajari bagaimana game biasa berinteraksi dengan Amazon GameLift.

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 GameLift plugin Amazon 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 GameLift realtime Amazon 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 IL2CPP ke dalam aplikasi seluler Anda yang menggunakan SDK Klien RTS dan mengikuti praktik terbaik dengan kerangka kerja. Anda juga sekarang dapat membangun Amazon GameLift 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 Server SDK, Amazon GameLift Local, dan plugin Unreal Engine.

Sorotan:

- Menambahkan dukungan untuk IL2CPP 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:

- Panduan GameLift Pengembang Amazon:
  - DescribePlayerSessions()
  - GetInstanceCertificate()

23 Maret 2021: Amazon GameLift 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 antrean sesi game. Buat topik Amazon Simple Notification Service (Amazon SNS) untuk mempublikasikan pemberitahuan acara, atau mengatur pelacakan peristiwa menggunakan Acara. CloudWatch

Sorotan:

- Untuk setiap antrean, 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 antrean tertentu.

Pelajari lebih lanjut:

- Panduan GameLift Pengembang Amazon:
  - Atur notifikasi kejadian untuk penempatan sesi game (baru)
  - Acara penempatan sesi game (baru)
- Referensi API (AWS SDK)
  - Parameter antrian sesi permainan baru `NotificationTarget` dan`CustomEventData`: GameSessionQueue,, CreateGameSessionQueueUpdateGameSessionQueue
- GameLiftForum Amazon

16 Maret 2021: Amazon GameLift menambahkan armada multi-wilayah, enam wilayah baru

Versi SDK yang diperbarui: AWS SDK 1.8.163

GameLift Hosting terkelola Amazon 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 GameLift multi-lokasi Amazon yang baru, Anda sekarang dapat menyiapkan satu armada untuk GameLift meng-host server game Anda di salah satu atau semua 20 Wilayah yang didukung Amazon (kecuali Wilayah Beijing). Fitur ini bertujuan untuk secara signifikan mengurangi pekerjaan yang diperlukan untuk menyiapkan dan memelihara sumber daya GameLift hosting Amazon secara global. Armada multi-lokasi dapat dibuat di AWS Wilayah berikut: `us-east-1` (Virginia N.), (Oregon), `us-west-2` (Frankfurt), `eu-central-1` (Irlandia), `eu-west-1` (Sydney), `ap-southeast-2` (Tokyo), dan `ap-northeast-1` (Seoul). `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. GameLiftHarga Amazon didasarkan pada jenis, lokasi, dan volume instans yang Anda gunakan. (Untuk informasi selengkapnya, lihat GameLiftharga Amazon.) AWS CloudFormation dukungan untuk armada multi-lokasi akan segera tersedia.

> **Note**
>
> Armada multi-lokasi tidak tersedia di Wilayah China. GameLiftSumber daya Amazon yang berada di Wilayah China tidak dapat berinteraksi atau digunakan oleh sumber daya di GameLift Wilayah Amazon lainnya.

Sorotan:

- Dengan armada multi-lokasi, tambahkan daftar lokasi jarak jauh secara eksplisit. Amazon GameLift 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 GameLift konsol Amazon, digabungkan untuk semua lokasi dalam armada atau dipecah berdasarkan 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,, DescribeEC2 UpdateFleetCapacityInstanceLimits,,, DescribeInstancesStopFleetActionsStartFleetActions
  - 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

- Panduan GameLift Pengembang Amazon:
  - Lokasi GameLift hosting Amazon (diperbarui)
  - Panduan desain GameLift armada Amazon (baru)

    Penskalaan kapasitas GameLift hosting Amazon (diperbarui)
  - Desain antrean sesi game (baru)
  - Meninjau detail armada (diperbarui)
- GameLiftForum Amazon

9 Februari 2021: Amazon GameLift memperluas dukungan untuk instans AMD, mandiri FlexMatch

Versi SDK yang diperbarui: AWS SDK 1.8.139

Rilis ini mencakup pembaruan berikut:

- Grup server game Amazon GameLift FleetiQ sekarang dapat dikonfigurasi untuk mengelola keluarga instans AMD C5a, M5a, dan R5a. Jenis instans Amazon EC2 yang didukung, seperti yang tercantum untuk GameServerGroup InstanceDefinition, sekarang mencakup 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 China (Beijing). AWS Lihat Ketersediaan fitur dan perbedaan implementasi di Tiongkok.
- Hosting game yang GameLift dikelola Amazon sekarang mendukung instans AMD di Wilayah China (Beijing), yang dioperasikan oleh Sinnet. Keluarga Instans AMD baru mencakup M5a dan R5a. Jenis instans EC2 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 sekarang GameLift FlexMatch dapat digunakan sebagai solusi perjodohan mandiri di Wilayah China (Beijing), yang dioperasikan oleh Sinnet. Pelanggan dapat membuat FlexMatch mak comblang di Wilayah Beijing dan mengonfigurasi FlexMatchModeparameter ke STANDALONE. Untuk informasi lebih lanjut tentang FlexMatch, baik dengan hosting GameLift terkelola Amazon atau dengan solusi GameLift hosting non-Amazon, di Panduan GameLift FlexMatch Pengembang Amazon.
- Saat menyiapkan notifikasi acara untuk Amazon GameLift FlexMatch, Anda sekarang dapat menetapkan topik Amazon SNS FIFO sebagai target notifikasi. Lihat informasi yang lebih lengkap di:
  - MatchmakingConfiguration NotificationTarget, Referensi GameLift API Amazon
  - Mengatur pemberitahuan FlexMatch acara, Panduan GameLift FlexMatch Pengembang Amazon
  - Memperkenalkan Amazon SNS FIFO - F irst-in-first-out Pub/Sub pesan, Blog Berita AWS

22 Desember 2020: SDK GameLift server Amazon mendukung Unreal Engine 4.25 dan Unity 2020

Versi SDK yang diperbarui: Amazon GameLift Server SDK 4.0.2, plugin Unreal versi 3.3.3

Versi terbaru dari Amazon GameLift 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 dari Amazon GameLift Server SDK di Amazon GameLift untuk memulai.

24 November 2020: Amazon GameLift FlexMatch sekarang tersedia untuk game yang dihosting di mana saja

Versi SDK yang diperbarui: AWS SDK 1.8.95

Amazon GameLift FlexMatch adalah layanan perjodohan yang dapat disesuaikan untuk game multipemain. Awalnya dirancang untuk pengguna hosting GameLift terkelola Amazon, 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 menggunakan

Amazon GameLift FleetiQ untuk hosting game di Amazon EC2 sekarang dapat menerapkan perjodohan dengan. FlexMatch

FlexMatch menyediakan algoritme perjodohan yang kuat dan bahasa aturan yang memberi Anda garis lintang lebar untuk menyesuaikan proses perjodohan sehingga pemain dicocokkan berdasarkan karakteristik pemain utama dan latensi yang dilaporkan. Selain itu, FlexMatch menawarkan alur kerja permintaan perjodohan yang mendukung fitur seperti pesta pemain, penerimaan pemain, dan pengisian ulang pertandingan. Saat Anda menggunakan FlexMatch hosting GameLift terkelola Amazon atau Server Realtime, mak comblang secara otomatis menggunakan Amazon GameLift 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 GameLift layanan Amazon, 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 GameLift terkelola Amazon atau sebagai perjodohan 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

Versi SDK yang diperbarui: AWS SDK 1.8.95

Daftar jenis instans Amazon EC2 yang didukung oleh Amazon GameLift sekarang mencakup tiga keluarga instans baru: C5a, M5a, dan R5a. Keluarga ini terdiri dari instans AMD yang dioptimalkan komputasi yang didukung oleh prosesor AMD EPYC yang berjalan pada frekuensi hingga 3,3. GHz. Instans AMD kompatibel dengan x86; game yang saat ini berjalan di Amazon GameLift 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 Central (Frankfurt), EU West (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
- Harga GameLift instans Amazon
- Instans Amazon EC2 yang menampilkan prosesor AMD EPYC
- GameLiftForum Amazon

11 November 2020: Pembaruan versi ke SDK GameLift server Amazon

Versi SDK yang diperbarui: Amazon GameLift 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 dari Amazon GameLift Server SDK di Amazon GameLift untuk 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 SDK.

- 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: Konfigurasikan mak comblang Anda untuk mengurutkan kumpulan tiket sebelum mengelompokkan tiket untuk evaluasi. 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 memperbarui SDK server

Versi SDK yang diperbarui: Amazon GameLift 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 F ProcessParameters pencocokan isi ulang.

Unduh versi terbaru dari Amazon GameLift Server SDK di Amazon GameLift untuk memulai.

27 Agustus 2020: Amazon GameLift FleetiQ untuk hosting game dengan Amazon EC2 (ketersediaan umum)

Versi SDK yang diperbarui: AWS SDK 1.8.36

Solusi Amazon GameLift FleetiQ untuk hosting game berbasis cloud berbiaya rendah di Amazon EC2 sekarang tersedia secara umum. Amazon GameLift FleetiQ memberi pengembang kemampuan untuk meng-host server game langsung di Instans Spot Amazon EC2 dengan mengoptimalkan kelangsungan hidup mereka untuk hosting game. Pengembang game dapat menggunakan Amazon GameLift 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 solusi Amazon GameLift FleetiQ:

- Operasi API baru `DescribeGameServerInstances` mengembalikan informasi, termasuk status, pada semua instans aktif untuk grup server game Amazon GameLift FleetiQ.
- 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 memperbarui SDK server untuk Unity dan Unreal Engine

Versi SDK yang diperbarui: Amazon GameLift Server SDK 4.0.0, Amazon Local 1.0.5 GameLift

Versi terbaru dari Amazon GameLift 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 Local versi 1.0.5 diperbarui untuk menguji integrasi yang menggunakan C# server SDK versi 4.0.0.

Unduh versi terbaru dari Amazon GameLift Server SDK di Amazon GameLift untuk memulai.

2 April 2020: Amazon GameLift FleetiQ tersedia untuk hosting game di EC2 (pratinjau publik)

Versi SDK yang diperbarui: AWS SDK 1.7.310

Fitur Amazon GameLift FleetiQ mengoptimalkan kelayakan 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 GameLift layanan Amazon 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 FleetiQ

19 Desember 2019: Peningkatan manajemen AWS sumber daya untuk GameLift sumber daya Amazon

Versi SDK yang diperbarui: AWS SDK 1.7.249

Anda sekarang dapat memanfaatkan alat manajemen AWS sumber daya dengan GameLift sumber daya Amazon. Secara khusus, semua GameLift sumber daya utama Amazon — 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, GameLift sumber daya Amazon 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 GameLift sumber daya Amazon, gunakan tindakan Amazon GameLift API `TagResource()``UntagResource()`, dan`ListTagsForResource()`.

Pelajari lebih lanjut:

- TagResourcedi Referensi GameLift API Amazon
- 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 China (Beijing)

Versi SDK yang diperbarui: AWS SDK 1.7.210

AWS CloudFormation template untuk Amazon GameLift

GameLift Sumber daya Amazon 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:

- Referensi jenis GameLift sumber daya Amazon di Panduan AWS CloudFormation Pengguna
- Mengelola sumber daya menggunakan AWS CloudFormationdi Panduan GameLift Pengembang Amazon

AWSGlosarium

Untuk AWS terminologi terbaru, lihat AWSglosarium di Referensi. Glosarium AWS

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.