aws

Guide du développeur

Amazon GameLift

Amazon GameLift: Guide du développeur

Table of Contents

Qu'est-ce qu'Amazon GameLift ?

Vous pouvez utiliser Amazon GameLift pour déployer, exploiter et faire évoluer des serveurs dédiés à faible coût dans le cloud pour les jeux multijoueurs basés sur des sessions. S'appuyant sur une infrastructure informatique AWS mondiale, Amazon GameLift vous aide à fournir des serveurs de jeu performants et fiables tout en adaptant dynamiquement votre utilisation des ressources pour répondre à la demande des joueurs du monde entier.

Utilisations d'Amazon GameLift

Amazon GameLift prend en charge les cas d'utilisation suivants et bien d'autres encore :

- Utilisez vos propres serveurs de jeu multijoueur personnalisés ou utilisez des serveurs en ready-to-go temps réel pour héberger vos jeux.
- Gérez des ressources d'hébergement à faible coût à l'aide des instances Spot Amazon Elastic Compute Cloud (Amazon EC2).
- Adaptez automatiquement la quantité de ressources d'hébergement dont votre jeu a besoin en fonction de son utilisation.
- Gérez vos ressources de calcul Amazon EC2 en un seul endroit à l'aide d'Amazon GameLift FleetIQ.
- Associez des joueurs dans des jeux multijoueurs avec Amazon GameLiftFlexMatch.
- Testez de manière itérative les versions de votre serveur de jeu et de votre client avec Amazon GameLiftAnywhere.
- Utilisez votre propre matériel tout en le gérant au même endroit avec Amazon GameLiftAnywhere.

> **Tip**
>
> Pour essayer l'hébergement de serveurs de GameLift jeux sur Amazon, consultezDébuter avec Amazon GameLift.

Démarrez avec les GameLift solutions Amazon

GameLiftSolutions Amazon pour les développeurs de jeux

- GameLiftHébergement Amazon pour serveurs personnalisés
- GameLiftHébergement Amazon avec serveurs en temps réel
- Amazon GameLift FleetIQ pour l'hébergement sur Amazon EC2
- Amazon GameLift FlexMatch pour le matchmaking
- Hébergement GameLift Anywhere matériel Amazon

GameLiftHébergement Amazon pour serveurs personnalisés

Amazon GameLift remplace le travail requis pour héberger vos propres serveurs de jeux personnalisés. Les fonctionnalités de mise à l'échelle automatique vous permettent d'éviter de payer pour plus de ressources que ce dont vous avez besoin. La mise à l'échelle automatique permet également de garantir que les nouveaux joueurs peuvent toujours accéder à des jeux avec un minimum d'attente.

Pour plus d'informations sur l'GameLifthébergement Amazon, consultezComment GameLift fonctionne Amazon.

Fonctions principales

- Utilisez les fonctionnalités GameLift de gestion d'Amazon, notamment la mise à l'échelle automatique, les files d'attente multi-sites et le placement des sessions de jeu.
- Déploiement de serveurs de jeux qui s'exécuteront sur les systèmes d'exploitation Amazon Linux ou Windows Server.
- Gestion des sessions de jeu et des sessions de joueur.
- Configurez un suivi de santé personnalisé pour les processus du serveur afin de détecter les problèmes et de résoudre les processus peu performants.
- Gérez vos ressources de jeu à l'aide AWS CloudFormation de modèles pour AmazonGameLift.

GameLiftHébergement Amazon avec serveurs en temps réel

Utilisez des serveurs en temps réel pour configurer des jeux qui ne nécessitent pas de serveurs de jeu personnalisés. Cette solution de serveur légère fournit des serveurs de jeu que vous pouvez configurer en fonction de votre jeu.

Pour plus d'informations sur l'GameLifthébergement Amazon avec des serveurs en temps réel, consultezIntégration de jeux aux serveurs Amazon GameLift Realtime.

Fonctions principales

- Utilisez les fonctionnalités GameLift de gestion d'Amazon, notamment la mise à l'échelle automatique, les files d'attente multi-sites et le placement des sessions de jeu.
- Utilisez les ressources GameLift d'hébergement d'Amazon et choisissez le type de matériel AWS informatique pour vos flottes.
- Profitez d'une infrastructure réseau complète pour l'interaction entre le client et le serveur du jeu.
- Bénéficiez des fonctionnalités intrinsèques du serveur de jeux avec une logique de serveur personnalisable.
- Effectuez des mises à jour en direct des configurations en temps réel et de la logique du serveur.

Amazon GameLift FleetIQ pour l'hébergement sur Amazon EC2

Utilisez Amazon GameLift FleetIQ pour travailler directement avec vos ressources d'hébergement dans Amazon EC2 et Amazon EC2 Auto Scaling. Cela permet de bénéficier des GameLift optimisations d'Amazon pour un hébergement de jeux résilient et peu coûteux. Cette solution s'adresse aux développeurs de jeux qui ont besoin d'une flexibilité supérieure à celle offerte par les GameLift solutions Amazon entièrement gérées.

Pour en savoir plus sur la façon dont Amazon GameLift FleetIQ fonctionne avec Amazon EC2 et EC2 Auto Scaling pour l'hébergement de jeux, consultez le Guide du développeur Amazon GameLift FleetIQ.

Fonctions principales

- Obtenez un équilibrage optimisé des instances Spot à l'aide de l'algorithme FleetIQ.
- Utilisez les fonctionnalités de routage des joueurs pour gérer efficacement les ressources de votre serveur de jeu et offrir une meilleure expérience aux joueurs lorsqu'ils rejoignent des parties.
- Adaptez automatiquement la capacité d'hébergement en fonction de l'utilisation des joueurs.
- Gérez directement les instances Amazon EC2 vous-même. Compte AWS
- Utilisez l'un des formats exécutables de serveur de jeu pris en charge, notamment Windows, Linux, les conteneurs et Kubernetes.

Amazon GameLift FlexMatch pour le matchmaking

Utilisez-le FlexMatch pour créer des ensembles de règles personnalisés afin de définir des matchs multijoueurs pour votre jeu. FlexMatchutilise des ensembles de règles pour comparer les joueurs compatibles pour chaque match et offrir aux joueurs une expérience multijoueur idéale.

Pour en savoir plusFlexMatch, consultez Qu'est-ce qu'Amazon GameLift FlexMatch ?

Fonctions principales

- Équilibrez la vitesse de création des matchs et la qualité des matchs.
- Associez des joueurs ou des équipes en fonction de caractéristiques définies.
- Définissez des règles pour placer les joueurs dans les matchs en fonction de la latence.

Hébergement GameLift Anywhere matériel Amazon

Utilisez Amazon GameLift Anywhere pour intégrer du matériel n'importe où dans votre environnement à votre hébergement de GameLift jeux Amazon. Vous pouvez intégrer Anywhere des flottes et des flottes EC2 dans des files d'attente de matchmaking et de sessions de jeu pour gérer le matchmaking et le placement des jeux sur votre matériel.

Pour plus d'informations sur les tests avecAnywhere, consultezTestez votre intégration à l'aide des GameLift Anywhere flottes Amazon. Pour plus d'informations sur la configuration d'une Anywhere flotte, consultezConfiguration des GameLift flottes Amazon.

Fonctions principales

- Effectuez des tests rapides et itératifs des versions de votre serveur de jeu et de votre client.
- Utilisez les GameLift outils Amazon prédéfinis pour déployer des jeux sur votre propre matériel.
- Utilisez le matériel le plus proche de vos joueurs, où que vous soyez.

Accès à Amazon GameLift

Utilisez ces outils pour travailler avec AmazonGameLift.

GameLiftSDK Amazon

> Les GameLift kits SDK Amazon contiennent les bibliothèques nécessaires pour communiquer avec Amazon à GameLift partir de vos clients de jeux, de vos serveurs de jeux et de vos services

de jeux. Pour plus d'informations, veuillez consulter Assistance au développement avec Amazon GameLift.

SDK GameLift du client Amazon Realtime

Le SDK Realtime Client permet à un client de jeu de se connecter au serveur Realtime, de rejoindre des sessions de jeu et de rester synchronisé avec les autres joueurs. Téléchargez le SDK pour en savoir plus sur les appels d'API avec l'API client Realtime Servers (C#).

GameLiftConsole Amazon

Utilisez le AWS Management Consolepour Amazon GameLift pour gérer les déploiements de vos jeux, configurer les ressources et suivre les statistiques d'utilisation et de performance des joueurs. La GameLift console Amazon fournit une interface graphique alternative à la gestion des ressources par programmation à l'aide du AWS Command Line Interface ()AWS CLI.

AWS CLI

Utilisez cet outil de ligne de commande pour appeler le AWS SDK, y compris l'GameLiftAPI Amazon. Pour plus d'informations sur l'utilisation deAWS CLI, consultez la section Mise AWS CLI en route du Guide de AWS Command Line Interface l'utilisateur.

Tarification pour Amazon GameLift

Amazon GameLift facture les instances en fonction de la durée d'utilisation et la bande passante en fonction de la quantité de données transférées. Pour obtenir la liste complète des frais et des prix d'AmazonGameLift, consultez la section GameLiftTarifs d'Amazon.

Pour plus d'informations sur le calcul du coût de l'hébergement de vos jeux ou du matchmaking avec Amazon GameLiftGénération d'estimations GameLift de prix Amazon, voir, qui décrit comment utiliser le AWS Pricing Calculator.

Comment GameLift fonctionne Amazon

Cette rubrique couvre les principaux composants de l'hébergement de jeux et décrit comment Amazon GameLift met vos serveurs de jeux multijoueurs à la disposition des joueurs.

Prêt à préparer votre jeu pour l'hébergement sur Amazon GameLift ? ConsultezFeuille de route d'hébergement GameLift géré par Amazon.

Composants clés

Pour configurer Amazon GameLift pour héberger votre jeu, vous devez utiliser les composants suivants. Le diagramme ci-dessous permet de Architecture de jeu avec Amazon géré GameLift visualiser les relations entre ces composants.

- Un serveur de jeu est le logiciel serveur de votre jeu qui s'exécute sur une flotte. Vous chargez la version ou le script de votre serveur de jeu sur Amazon GameLift et vous le dites à AmazonGameLift. Lorsque vous utilisez Amazon GameLift Anywhere ou Amazon GameLift FleetIQ, vous chargez la version de votre serveur de jeu directement sur la ressource informatique.
- Une session de jeu est une partie en cours avec des joueurs. Vous définissez les caractéristiques de base d'une session de jeu, comme sa durée de vie ou le nombre de joueurs. Les joueurs se connectent ensuite au serveur de jeu pour rejoindre une session de jeu.
- Un client de jeu est le logiciel de votre jeu qui s'exécute sur le dispositif d'un joueur. Un client de jeu se connecte à un serveur de jeu via des services principaux pour rejoindre une session de jeu, en fonction des informations de connexion qu'il reçoit d'AmazonGameLift.
- Les services dorsaux sont des services supplémentaires personnalisés qui gèrent les tâches liées à AmazonGameLift. La meilleure pratique consiste à faire en sorte que vos services principaux gèrent toutes les communications entre le client du jeu et AmazonGameLift.

Hébergement de serveurs de jeux

Avec AmazonGameLift, vous pouvez héberger vos serveurs de jeux de trois manières différentes : Amazon géréGameLift, Amazon GameLift FleetIQ et Amazon. GameLift Anywhere Pour plus d'informations sur Amazon GameLift FleetIQ, consultez Qu'est-ce qu'Amazon GameLift FleetIQ ?

Vous pouvez concevoir une flotte en fonction des besoins de votre jeu. Pour plus d'informations sur la conception d'une flotte, consultezGuide GameLift de conception de flotte Amazon.

Amazon géré GameLift

Avec Amazon géréGameLift, vous pouvez héberger vos serveurs de jeu sur les ressources informatiques GameLift virtuelles d'Amazon, appelées instances. Configurez vos ressources d'hébergement en créant un parc d'instances et en les déployant pour faire fonctionner vos serveurs de jeu.

Amazon GameLift Anywhere

Avec Amazon GameLiftAnywhere, vous pouvez héberger vos serveurs de jeu sur des ordinateurs que vous gérez. Configurez vos ressources d'hébergement en créant une Anywhere flotte qui fait référence à votre calcul.

Alias de flotte

Un alias est une désignation que vous pouvez transférer d'une flotte à une autre, ce qui en fait un moyen pratique d'avoir un emplacement de flotte générique. Vous pouvez utiliser un alias pour changer de client de jeu d'une flotte à une autre sans changer de client de jeu. Vous pouvez également créer un alias de terminal vers lequel vous pointez vers du contenu.

Sessions de jeu en cours

Une fois que vous avez déployé la version de votre serveur de jeu sur une flotte et qu'Amazon a GameLift lancé des processus de serveur de jeu sur chaque instance, la flotte peut héberger des sessions de jeu. Amazon GameLift lance de nouvelles sessions de jeu lorsque le service client de votre jeu envoie une demande de placement au service principal ou à AmazonGameLift.

Placement des sessions de jeu et algorithme FleetIQ

Les files d'attente utilisent l'algorithme FleetIQ pour sélectionner un serveur de jeu disponible pour héberger une nouvelle session de jeu. L'élément clé pour le placement des sessions de jeu est la file d'attente des sessions de GameLift jeu Amazon. Vous attribuez à une file d'attente de sessions de jeu une liste de flottes, qui détermine où la file d'attente peut placer les sessions de jeu. Pour plus d'informations sur les files d'attente des sessions de jeu et sur la façon de les concevoir pour votre jeu, consultezConception d'une file d'attente de sessions de jeu.

Connexions des joueurs aux jeux

Dans le cadre du processus de placement d'une session de jeu, la file d'attente ou la session de jeu invite le serveur de jeu sélectionné à démarrer une nouvelle session de jeu. Le serveur de jeu répond à l'invite et signale à Amazon GameLift lorsqu'il est prêt à accepter les connexions des joueurs. Amazon fournit GameLift ensuite les informations de connexion au service principal ou au service client du jeu. Les clients de votre jeu utilisent ces informations pour se connecter directement à la session de jeu et commencer à jouer.

Dimensionnement de la capacité d'une flotte

Lorsqu'une flotte est active et prête à héberger des sessions de jeu, vous pouvez ajuster la capacité de votre flotte pour répondre à la demande des joueurs. Nous vous recommandons de trouver un

équilibre entre le fait que tous les nouveaux joueurs trouvent rapidement une partie et le fait de dépenser trop de ressources inutilisées.

Amazon GameLift fournit un outil de mise à l'échelle automatique très efficace. Vous pouvez également définir manuellement la capacité de votre flotte. Pour plus d'informations, veuillez consulter Élargir la capacité GameLift d'hébergement d'Amazon.

Auto scaling (Mise à l'échelle automatique)

Amazon GameLift propose deux méthodes de mise à l'échelle automatique :

- Mise à l'échelle automatique basée sur les cibles
- Mise à l'échelle automatique avec des politiques basées sur des règles

Fonctionnalités de mise à l'échelle supplémentaires

- Protection des sessions de jeu : GameLift empêchez Amazon de mettre fin aux sessions de jeu qui accueillent des joueurs actifs lors d'un événement de réduction de la taille.
- Limites de dimensionnement : contrôlez l'utilisation globale des instances en définissant des limites minimales et maximales pour le nombre d'instances dans un parc.
- Suspension du dimensionnement automatique — Suspendez le dimensionnement automatique au niveau de l'emplacement de la flotte sans modifier ni supprimer vos politiques de dimensionnement automatique.
- Métriques de mise à l'échelle : suivez l'historique de la capacité et des événements de mise à l'échelle d'une flotte.

Surveillance d'Amazon GameLift

Lorsque votre flotte est opérationnelle, Amazon GameLift collecte diverses informations pour vous aider à surveiller les performances des serveurs de jeu que vous déployez. Vous pouvez utiliser ces informations pour optimiser votre utilisation des ressources, résoudre des problèmes et mieux comprendre comment les joueurs sont actifs dans vos jeux. Amazon GameLift collecte les informations suivantes :

- Détails de la flotte, de l'emplacement, de la session de jeu et de la session des joueurs
- Métriques d'utilisation
- État des processus du serveur

- Journaux des sessions de jeu

Pour plus d'informations sur la surveillance sur AmazonGameLift, consultezSurveillance d'Amazon GameLift.

Utilisation d'autres AWS ressources

Vos serveurs de jeu et vos applications peuvent communiquer avec d'autres AWS ressources. Par exemple, vous pouvez utiliser un ensemble de services Web pour l'authentification des joueurs ou les réseaux sociaux. Pour que vos serveurs de jeu puissent accéder aux AWS ressources que vous Compte AWS gérez, autorisez explicitement Amazon GameLift à accéder à vos AWS ressources.

Amazon GameLift propose plusieurs options pour gérer ce type d'accès. Pour plus d'informations, veuillez consulter Communiquez avec les autres AWS ressources de vos flottes.

Comment les joueurs se connectent aux jeux

Une session de jeu est une instance de votre jeu exécutée sur AmazonGameLift. Pour jouer à votre jeu, un joueur peut soit rechercher et rejoindre une session de jeu existante, soit créer une nouvelle session de jeu et la rejoindre. Un joueur rejoint le jeu en créant une session de joueur pour la session de jeu. Si la session de jeu est ouverte aux joueurs, Amazon leur GameLift réserve un emplacement et leur fournit des informations de connexion. Le joueur peut alors se connecter à la session de jeu et profiter de l'emplacement réservé.

Pour des informations détaillées sur la création et la gestion de sessions de jeu et de sessions de joueurs avec des serveurs de jeu personnalisés, consultezAjoutez Amazon GameLift à votre client de jeu. Pour plus d'informations sur la connexion des joueurs aux serveurs en temps réel, consultezIntégration d'un client de jeu pour les serveurs en temps réel.

Amazon GameLift propose plusieurs fonctionnalités liées aux sessions de jeu et aux joueurs.

Organisez des sessions de jeu en utilisant les meilleures ressources disponibles sur plusieurs sites

> Choisissez parmi plusieurs options lorsque vous configurez la façon dont Amazon GameLift sélectionne les ressources pour héberger de nouvelles sessions de jeu. Si vous gérez des flottes à plusieurs endroits, vous pouvez créer des files d'attente de sessions de jeu qui permettent de placer de nouvelles sessions de jeu sur n'importe quelle flotte, quel que soit son emplacement.

Contrôlez l'accès des joueurs aux sessions de jeu

Configurez les sessions de jeu pour autoriser ou refuser les demandes de participation de nouveaux joueurs, quel que soit le nombre de joueurs connectés.

Utiliser des données de jeu et de joueur personnalisées

Ajoutez des données personnalisées aux objets de session de jeu et aux objets de session des joueurs. Amazon GameLift transmet les données de session de jeu à un serveur de jeu lors du démarrage d'une nouvelle session de jeu. Amazon GameLift transmet les données de session des joueurs au serveur de jeu lorsqu'un joueur se connecte à la session de jeu.

Filtrer et trier les sessions de jeu disponibles

Utilisez la recherche et le tri par session pour trouver le meilleur match possible pour un joueur potentiel, ou laissez le joueur choisir parmi une liste de sessions de jeu disponibles. Utilisez la recherche et le tri des sessions pour trouver des sessions de jeu en fonction de leurs caractéristiques. Vous pouvez également rechercher et trier selon vos propres données de jeu personnalisées.

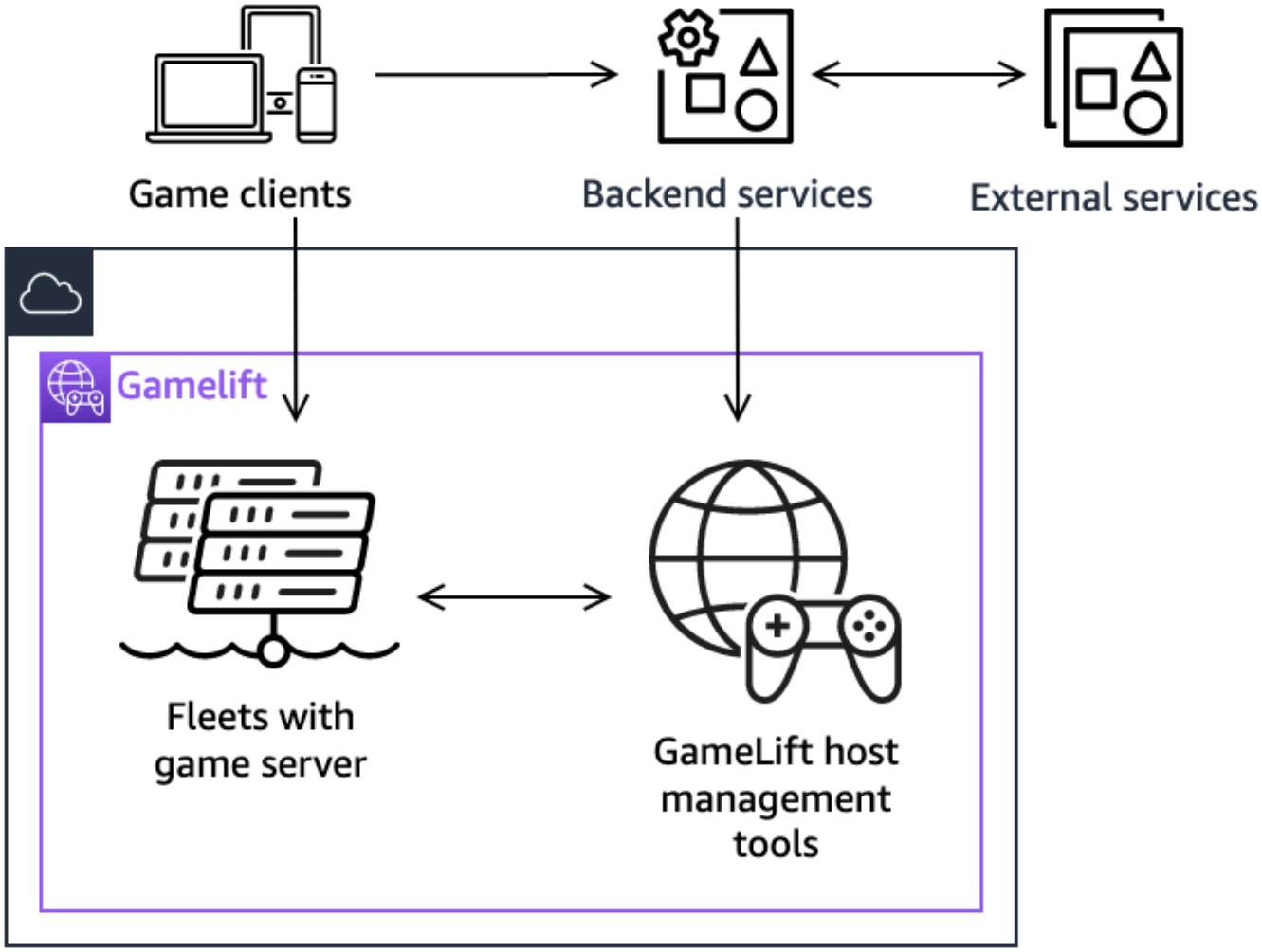
Suivez les données d'utilisation du jeu et des joueurs

Enregistrez automatiquement les journaux des sessions de jeu terminées. Vous pouvez configurer le stockage des journaux lorsque vous intégrez Amazon GameLift à vos serveurs de jeu. Pour plus d'informations, veuillez consulter Enregistrement des messages du serveur dans Amazon GameLift.

Utilisez la GameLift console Amazon pour consulter des informations détaillées sur les sessions de jeu, notamment les métadonnées de session, les paramètres et les données de session des joueurs. Pour plus d'informations, consultez Afficher les données sur les sessions de jeu et les sessions des joueurs et Métriques.

Architecture de jeu avec Amazon géré GameLift

Le schéma suivant illustre les composants clés d'une architecture de jeu hébergée à l'aide de la GameLift solution Amazon gérée.



Les principaux composants de cette architecture sont les suivants :

Clients du jeu

Pour rejoindre un jeu hébergé sur Amazon GameLift, votre client de jeu doit d'abord trouver une session de jeu disponible. Le client du jeu recherche des sessions de jeu existantes, demande le matchmaking ou démarre une nouvelle session de jeu en communiquant avec Amazon GameLift via un service principal. Le service principal envoie des demandes à Amazon et GameLift, en réponse, le service reçoit des informations de session de jeu, qu'il transmet au client du jeu. Le client de jeu se connecte ensuite au serveur de jeu. Pour plus d’informations, consultez Préparation de jeux pour Amazon GameLift.

Services de backend

Un service principal gère la communication entre les clients du jeu et Amazon GameLift en appelant les opérations de l'API du GameLift service Amazon dans le AWS SDK. Vous pouvez également utiliser les services de backend pour d'autres tâches spécifiques au jeu, telles que l'authentification et l'autorisation des joueurs, l'inventaire ou le contrôle des devises. Pour plus d'informations, consultez Concevez le service client de votre jeu.

Services externes

Votre jeu peut s'appuyer sur un service externe, par exemple pour valider un abonnement. Un service externe peut transmettre des informations à vos serveurs de jeu via un service principal et Amazon GameLift.

Serveurs de jeux

Vous téléchargez le logiciel de votre serveur de jeu sur Amazon GameLift, GameLift puis Amazon le déploie sur des machines d'hébergement pour héberger des sessions de jeu et accepter les connexions entre joueurs. Les serveurs de jeu communiquent avec Amazon GameLift pour démarrer des sessions de jeu, valider les nouveaux joueurs connectés et signaler l'état des sessions de jeu, des connexions des joueurs et des ressources disponibles.

Les serveurs de jeux personnalisés communiquent avec Amazon GameLift à l'aide GameLift du SDK Amazon Server. Les clients du jeu se connectent directement à un serveur de jeu après avoir reçu les informations de connexion d'Amazon GameLift via un service principal. Pour plus d'informations, consultez Intégrez des jeux à des serveurs de jeux personnalisés.

Les serveurs en temps réel sont des serveurs de jeu qui exécutent votre script personnalisé. Lorsque vous rejoignez un jeu, un client de jeu se connecte directement à un serveur en temps réel à l'aide du SDK du client en temps réel. Pour plus d'informations, consultez Intégration de jeux aux serveurs Amazon GameLift Realtime.

Outils de gestion des hôtes

Lors de la configuration et de la gestion des ressources d'hébergement, les propriétaires de jeux utilisent des outils de gestion de l'hébergement pour gérer les versions ou les scripts des serveurs de jeu, les flottes, le matchmaking et les files d'attente. L' GameLift outil Amazon intégré au AWS SDK et à la console vous permet de gérer vos ressources d'hébergement de différentes manières. Vous pouvez accéder à distance à n'importe quel serveur de jeu individuel pour résoudre les problèmes.

Configuration

Obtenez de l'aide pour configurer votre utilisation Compte AWS d'Amazon GameLift pour héberger vos jeux multijoueurs.

> **Tip**
>
> Pour essayer l'hébergement de serveurs de GameLift jeux sur Amazon, consultezDébuter avec Amazon GameLift.

Rubriques

Configurez un Compte AWS

Pour commencer à utiliser Amazon GameLift, créez et configurez votre Compte AWS. La création d'un Compte AWS. Cette section explique comment créer votre compte, configurer vos utilisateurs et configurer les autorisations.

Rubriques

Inscrivez-vous pour un Compte AWS

Si vous n'en avez pas Compte AWS, procédez comme suit pour en créer un.

Pour vous inscrire à un Compte AWS

1. Ouvrez https://portal.aws.amazon.com/billing/signup.
2. Suivez les instructions en ligne.

   Dans le cadre de la procédure d'inscription, vous recevrez un appel téléphonique et vous saisirez un code de vérification en utilisant le clavier numérique du téléphone.

   Lorsque vous vous inscrivez à un Compte AWS, un Utilisateur racine d'un compte AWSest créé. Par défaut, seul l'utilisateur racine a accès à l'ensemble des Services AWS et des ressources de ce compte. Pour des raisons de sécurité, attribuez un accès administratif à un utilisateur et utilisez uniquement l'utilisateur root pour effectuer les tâches nécessitant un accès utilisateur root.

AWS vous envoie un e-mail de confirmation une fois le processus d'inscription terminé. Vous pouvez afficher l'activité en cours de votre compte et gérer votre compte à tout moment en accédant à https://aws.amazon.com/ et en choisissant Mon compte.

Création d'un utilisateur doté d'un accès administratif

Une fois que vous vous êtes inscrit à un utilisateur administratif Compte AWS, que vous Utilisateur racine d'un compte AWS l'avez sécurisé AWS IAM Identity Center, que vous l'avez activé et que vous en avez créé un, afin de ne pas utiliser l'utilisateur root pour les tâches quotidiennes.

Sécurisez votre Utilisateur racine d'un compte AWS

1. Connectez-vous en AWS Management Consoletant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.

   Pour obtenir de l'aide pour vous connecter en utilisant l'utilisateur racine, consultez Connexion en tant qu'utilisateur racine dans le Guide de l'utilisateur Connexion à AWS .
2. Activez l'authentification multifactorielle (MFA) pour votre utilisateur racine.

Pour obtenir des instructions, voir Activer un périphérique MFA virtuel pour votre utilisateur Compte AWS root (console) dans le guide de l'utilisateur IAM.

Création d'un utilisateur doté d'un accès administratif

1. Activez IAM Identity Center.

   Pour obtenir des instructions, consultez Activation d' AWS IAM Identity Center dans le Guide de l'utilisateur AWS IAM Identity Center .
2. Dans IAM Identity Center, accordez un accès administratif à un utilisateur.

   Pour un didacticiel sur l'utilisation du Répertoire IAM Identity Center comme source d'identité, voir Configurer l'accès utilisateur par défaut Répertoire IAM Identity Center dans le Guide de AWS IAM Identity Center l'utilisateur.

Connectez-vous en tant qu'utilisateur disposant d'un accès administratif

- Pour vous connecter avec votre utilisateur IAM Identity Center, utilisez l'URL de connexion qui a été envoyée à votre adresse e-mail lorsque vous avez créé l'utilisateur IAM Identity Center.

  Pour obtenir de l'aide pour vous connecter en utilisant un utilisateur d'IAM Identity Center, consultez la section Connexion au portail AWS d'accès dans le guide de l'Connexion à AWS utilisateur.

Attribuer l'accès à des utilisateurs supplémentaires

1. Dans IAM Identity Center, créez un ensemble d'autorisations conforme aux meilleures pratiques en matière d'application des autorisations du moindre privilège.

   Pour obtenir des instructions, voir Création d'un ensemble d'autorisations dans le guide de AWS IAM Identity Center l'utilisateur.
2. Affectez des utilisateurs à un groupe, puis attribuez un accès d'authentification unique au groupe.

   Pour obtenir des instructions, voir Ajouter des groupes dans le guide de AWS IAM Identity Center l'utilisateur.

Gérer les autorisations des utilisateurs pour Amazon GameLift

Créez des utilisateurs supplémentaires ou étendez les autorisations d'accès aux utilisateurs existants selon les besoins de vos GameLift ressources Amazon. La meilleure pratique (meilleures pratiques de sécurité dans IAM) consiste à appliquer des autorisations de moindre privilège à tous les utilisateurs. Pour obtenir des conseils sur la syntaxe des autorisations, voirExemples d'autorisations IAM pour Amazon GameLift.

Suivez les instructions ci-dessous pour définir les autorisations utilisateur en fonction de la façon dont vous gérez les utilisateurs de votre AWS compte.

Pour activer l’accès, ajoutez des autorisations à vos utilisateurs, groupes ou rôles :

- Utilisateurs et groupes dans AWS IAM Identity Center :

  Créez un jeu d’autorisations. Suivez les instructions de la rubrique Création d’un jeu d’autorisations du Guide de l’utilisateur AWS IAM Identity Center .
- Utilisateurs gérés dans IAM par un fournisseur d’identité :

  Créez un rôle pour la fédération d’identité. Pour plus d’informations, voir la rubrique Création d’un rôle pour un fournisseur d’identité tiers (fédération) du Guide de l’utilisateur IAM.
- Utilisateurs IAM :
  - Créez un rôle que votre utilisateur peut assumer. Suivez les instructions de la rubrique Création d’un rôle pour un utilisateur IAM du Guide de l’utilisateur IAM.
  - (Non recommandé) Attachez une politique directement à un utilisateur ou ajoutez un utilisateur à un groupe d’utilisateurs. Suivez les instructions de la rubrique Ajout d’autorisations à un utilisateur (console) du Guide de l’utilisateur IAM.

Lorsque vous travaillez avec des utilisateurs IAM, il est recommandé de toujours associer des autorisations à des rôles ou à des groupes d'utilisateurs, et non à des utilisateurs individuels.

Configurer l'accès programmatique pour les utilisateurs

Les utilisateurs ont besoin d'un accès programmatique s'ils souhaitent interagir avec AWS l'extérieur du AWS Management Console. La manière d'accorder un accès programmatique dépend du type d'utilisateur qui y accède AWS.

Pour accorder aux utilisateurs un accès programmatique, choisissez l’une des options suivantes.

Quel utilisateur a besoin d'un accès programmatique ?	Pour	Par
Identité de la main-d'œuvre (Utilisateurs gérés dans IAM Identity Center)	Utilisez des informations d'identification temporaires pour signer les demandes programmatiques adressées aux AWS CLI AWS SDK ou AWS aux API.	Suivez les instructions de l'interface que vous souhaitez utiliser. • Pour le AWS CLI, voir Configuration du AWS CLI à utiliser AWS IAM Identity Center dans le guide de AWS Command Line Interface l'utilisateur. • Pour les AWS SDK, les outils et les AWS API, consultez la section Authentification IAM Identity Center dans le Guide de référence AWS des SDK et des outils.
IAM	Utilisez des informations d'identification temporaires pour signer les demandes programmatiques adressées aux AWS CLI AWS SDK ou AWS aux API.	Suivez les instructions de la section Utilisation d'informations d'identification temporaires avec AWS les ressources du Guide de l'utilisateur IAM.
IAM	(Non recommandé) Utilisez des informations d'identification à long terme pour signer les AWS CLI demandes programmatiques adressées aux AWS SDK ou AWS aux API.	Suivez les instructions de l'interface que vous souhaitez utiliser. • Pour le AWS CLI, voir Authentification à l'aide des informations d'identification utilisateur IAM dans le

Quel utilisateur a besoin d'un accès programmatique ?	Pour	Par
		Guide de l'AWS Command Line Interface utilisateur. • Pour les AWS SDK et les outils, voir Authentifier à l'aide d'informations d'identification à long terme dans le Guide de AWS référence des SDK et des outils. • Pour les AWS API, consultez la section Gestion des clés d'accès pour les utilisateurs IAM dans le guide de l'utilisateur IAM.

Si vous utilisez des clés d'accès, consultez la section Meilleures pratiques de gestion des clés AWS d'accès.

Configurez l'accès programmatique à votre jeu

La plupart des jeux utilisent des services principaux pour communiquer avec Amazon à GameLift l'aide des AWS SDK. Par exemple, vous utilisez un service principal (agissant pour le compte des clients du jeu) pour demander des sessions de jeu, placer des joueurs dans des jeux et effectuer d'autres tâches. Ces services ont besoin d'un accès programmatique et d'informations de sécurité pour authentifier les appels aux API des GameLift services Amazon.

Pour Amazon GameLift, vous gérez cet accès en créant un utilisateur joueur dans AWS Identity and Access Management (IAM). Gérez les autorisations des utilisateurs des joueurs via l'une des options suivantes :

- Créez un rôle IAM avec les autorisations d'utilisateur du joueur et autorisez celui-ci à assumer le rôle en cas de besoin. Le service principal doit inclure du code permettant d'assumer ce rôle avant d'envoyer des demandes à Amazon GameLift. Conformément aux meilleures pratiques de sécurité, les rôles fournissent un accès temporaire limité. Vous pouvez utiliser des rôles pour les

charges de travail exécutées sur AWS des ressources (rôles IAM) ou en dehors de AWS (IAM Roles Anywhere).

- Créez un groupe d'utilisateurs IAM avec des autorisations d'utilisateur joueur et ajoutez votre utilisateur joueur au groupe. Cette option fournit à votre joueur des informations d'identification à long terme, que le service principal doit stocker et utiliser lors de la communication avec Amazon GameLift.

Pour la syntaxe de la politique d'autorisation, voirExemples d'autorisations utilisateur pour les joueurs.

Pour plus d'informations sur la gestion des autorisations destinées à être utilisées par un workload, voir Identités IAM : informations d'identification temporaires dans IAM.

Exemples d'autorisations IAM pour Amazon GameLift

Utilisez la syntaxe de ces exemples pour définir des autorisations AWS Identity and Access Management (IAM) pour les utilisateurs qui ont besoin d'accéder aux GameLift ressources Amazon. Pour plus d'informations sur la gestion des autorisations des utilisateurs, consultezGérer les autorisations des utilisateurs pour Amazon GameLift. Lorsque vous gérez les autorisations des utilisateurs en dehors de l'IAM Identity Center, il est recommandé de toujours associer les autorisations aux rôles IAM ou aux groupes d'utilisateurs, et non aux utilisateurs individuels.

Si vous utilisez Amazon GameLift FleetIQ comme solution autonome, consultez Configurer votre Compte AWS pour Amazon FleetIQ. GameLift

Exemples d'autorisations d'administrateur

Ces exemples donnent à un utilisateur un accès complet à la gestion des ressources d'hébergement de GameLift jeux Amazon.

Example Syntaxe pour les autorisations relatives GameLift aux ressources Amazon

L'exemple suivant étend l'accès à toutes les GameLift ressources Amazon.

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

```
  }
}
```

Example Syntaxe pour les autorisations relatives aux GameLift ressources Amazon avec prise en charge des régions qui ne sont pas activées par défaut

L'exemple suivant étend l'accès à toutes les GameLift ressources et AWS régions Amazon qui ne sont pas activées par défaut. Pour plus d'informations sur les régions qui ne sont pas activées par défaut et sur la façon de les activer, consultez la section Gestion Régions AWS dans le Références générales AWS.

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

Example Syntaxe pour les GameLift ressources et les **`PassRole`** autorisations Amazon

L'exemple suivant étend l'accès à toutes les GameLift ressources Amazon et permet à un utilisateur de transmettre un rôle de service IAM à AmazonGameLift. Un rôle de service donne à Amazon une capacité GameLift limitée à accéder à d'autres ressources et services en votre nom, comme décrit dansConfigurer un rôle de service IAM pour Amazon GameLift. Par exemple, lorsque vous répondez à une `CreateBuild` demande, Amazon GameLift doit accéder à vos fichiers de compilation dans un compartiment Amazon S3. Pour plus d'informations sur cette `PassRole` action, voir IAM : Transmettre un rôle IAM à un AWS service spécifique dans le Guide de l'utilisateur IAM.

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

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

Exemples d'autorisations utilisateur pour les joueurs

Ces exemples permettent à un service principal ou à une autre entité d'effectuer des appels d'API vers l'GameLiftAPI Amazon. Ils couvrent les scénarios courants pour la gestion des sessions de jeu, des sessions de joueurs et du matchmaking. Pour en savoir plus, consultez Configurez l'accès programmatique à votre jeu.

Example Syntaxe pour les autorisations de placement des sessions de jeu

L'exemple suivant étend l'accès aux GameLift API Amazon qui utilisent des files d'attente de placement de sessions de jeu pour créer des sessions de jeu et gérer les sessions des joueurs.

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

Example Syntaxe pour les autorisations de matchmaking

L'exemple suivant étend l'accès aux GameLift API Amazon qui gèrent les activités FlexMatch de matchmaking. FlexMatchassocie les joueurs à des sessions de jeu nouvelles ou existantes et initie le placement des sessions de jeu pour les jeux hébergés sur AmazonGameLift. Pour en savoir plusFlexMatch, consultez Qu'est-ce qu'Amazon GameLift FlexMatch ?

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

Example Syntaxe pour les autorisations de placement manuel des sessions de jeu

L'exemple suivant étend l'accès aux GameLift API Amazon qui créent manuellement des sessions de jeu et des sessions pour les joueurs sur des flottes spécifiques. Ce scénario prend en charge les jeux qui n'utilisent pas de files d'attente de placement, tels que les jeux qui permettent aux joueurs de s'inscrire en choisissant parmi une liste de sessions de jeu disponibles (méthode « 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": "*"
  }
}
```

Configurer un rôle de service IAM pour Amazon GameLift

Certaines GameLift fonctionnalités d'Amazon nécessitent que vous étendiez un accès limité aux AWS ressources que vous possédez. Vous pouvez le faire en créant un rôle AWS Identity and Access Management (IAM). Un rôle IAM est une identité IAM que vous pouvez créer dans votre compte et qui dispose d'autorisations spécifiques. Un rôle IAM est similaire à un utilisateur IAM, car il s'agit d'une identité AWS avec des politiques d'autorisation qui déterminent ce que l'identité peut et ne peut pas faire dans AWS. En revanche, au lieu d'être associé de manière unique à une personne, un rôle est conçu pour être assumé par tout utilisateur qui en a besoin. En outre, un rôle ne dispose pas d'informations d'identification standard à long terme comme un mot de passe ou des clés d'accès associées. Au lieu de cela, lorsque vous adoptez un rôle, il vous fournit des informations d'identification de sécurité temporaires pour votre session de rôle.

Cette rubrique explique comment créer un rôle que vous pouvez utiliser avec vos flottes GameLift gérées par Amazon. Si vous utilisez Amazon GameLift FleetIQ pour optimiser l'hébergement de jeux sur vos instances Amazon Elastic Compute Cloud (Amazon EC2), consultez Configurer votre compte pour Amazon FleetIQ. Compte AWS GameLift

Dans la procédure suivante, créez un rôle avec une politique d'autorisation personnalisée et une politique de confiance qui autorise Amazon GameLift à assumer le rôle.

Création d'un rôle IAM personnalisé

Étape 1 : créer une politique d'autorisation.

Pour utiliser l'éditeur de politique JSON afin de créer une politique

1. Connectez-vous à la AWS Management Console et ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de gauche, sélectionnez Policies (Politiques).

   Si vous sélectionnez Politiques pour la première fois, la page Bienvenue dans les politiques gérées s'affiche. Sélectionnez Mise en route.
3. En haut de la page, sélectionnez Créer une politique.

4. Dans la section Éditeur de politiques, choisissez l'option JSON.
5. Saisissez ou collez un document de politique JSON. Pour de plus amples informations sur le langage de la stratégie IAM, consultez la référence de politique JSON IAM.
6. Résolvez les avertissements de sécurité, les erreurs ou les avertissements généraux générés durant la validation de la politique, puis choisissez Suivant.

   > **Note**
   >
   > Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'éditeur visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour de plus amples informations, consultez la page Restructuration de politique dans le Guide de l'utilisateur IAM.

7. (Facultatif) Lorsque vous créez ou modifiez une politique dans AWS Management Console, vous pouvez générer un modèle de politique JSON ou YAML que vous utilisez dans les modèles AWS CloudFormation.

   Pour ce faire, dans l'éditeur de politiques, sélectionnez Actions, puis sélectionnez Générer CloudFormation un modèle. Pour en savoir plus sur AWS CloudFormation, consultez Référence des types de ressource AWS Identity and Access Management dans le Guide de l'utilisateur AWS CloudFormation.
8. Lorsque vous avez fini d'ajouter des autorisations à la politique, choisissez Suivant.
9. Sur la page Vérifier et créer, tapez un Nom de politique et une Description (facultative) pour la politique que vous créez. Vérifiez les Autorisations définies dans cette politique pour voir les autorisations accordées par votre politique.
10. (Facultatif) Ajoutez des métadonnées à la politique en associant les balises sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, consultez la rubrique Balisage des ressources IAM dans le Guide de l'utilisateur IAM.
11. Choisissez Create policy (Créer une politique) pour enregistrer votre nouvelle politique.

Étape 2 : Créez un rôle qu'Amazon GameLift peut assumer.

1. Dans le volet de navigation de la console IAM, choisissez Rôles, puis Créer un rôle.
2. Sur la page Sélectionner une entité de confiance, choisissez l'option Politique de confiance personnalisée. Cette sélection ouvre l'éditeur de politique de confiance personnalisée.

3. Remplacez la syntaxe JSON par défaut par la suivante, puis choisissez Next pour continuer.

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

4. Sur la page Ajouter des autorisations, recherchez et sélectionnez la politique d'autorisation que vous avez créée à l'étape 1. Choisissez Next (Suivant) pour continuer.
5. Sur la page Nom, révision et création, entrez un nom de rôle et une description (facultatif) pour le rôle que vous créez. Passez en revue les entités de confiance et les autorisations ajoutées.
6. Choisissez Créer un rôle pour enregistrer votre nouveau rôle.

Syntaxe de la politique d'autorisation

- Autorisations permettant GameLift à Amazon d'assumer le rôle de service

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

- Autorisations d'accès aux AWS régions qui ne sont pas activées par défaut

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

Assistance au développement avec Amazon GameLift

Amazon GameLift fournit un ensemble de SDK que vous pouvez utiliser avec vos solutions d'hébergement de jeux gérés. Utilisez GameLift les SDK Amazon pour ajouter les fonctionnalités nécessaires aux serveurs de jeux multijoueurs, aux clients de jeux et aux services de jeux qui doivent interagir avec le service GameLift d'hébergement Amazon.

Pour obtenir les dernières informations sur les versions du GameLift SDK Amazon et la compatibilité avec le SDK, consultez. Notes de GameLift publication d'Amazon

Pour les serveurs de jeu personnalisés

Créez et déployez des serveurs de jeu personnalisés 64 bits avec le SDK Amazon GameLift Server. Les serveurs de jeu intégrés au SDK du serveur et déployés pour l'hébergement peuvent communiquer avec le GameLift service Amazon pour démarrer et gérer des sessions de jeu. Pour plus d'informations sur l'intégration du SDK du serveur, consultez les rubriques dePréparation de jeux pour Amazon GameLift.

Systèmes d'exploitation pour le développement

- Windows
- Linux

Langages de programmation pris en charge

Amazon GameLift fournit le SDK du serveur pour les langues suivantes. Téléchargez les SDK pour serveurs. Pour des informations détaillées spécifiques à chaque version, consultez les fichiers Readme inclus dans chaque package.

- SDK du serveur C++
  - Référence du SDK
  - Intégration du SDK
- SDK du serveur C# (les versions peuvent prendre en charge .NET 4 et .NET 6)
  - Référence du SDK
  - Intégration du SDK
- Go
  - Référence du SDK
  - Intégration du SDK

Moteurs de jeu compatibles

Utilisez des SDK spécifiques au langage avec tous les moteurs prenant en charge les bibliothèques C++, C# ou Go. En outre, Amazon GameLift fournit les plugins suivants pour les moteurs de jeu : Téléchargez les GameLift plugins Amazon

- Unité
  - Le plugin SDK du serveur C# pour Unity est un plugin léger avec des bibliothèques prédéfinies que vous pouvez installer à l'aide du gestionnaire de packages Unity. Utilisez ce plugin avec les versions Unity suivantes : 2020.3 LTS, 2021.3 LTS et 2022.3 LTS pour Windows et Mac OS. Il prend en charge les profils .NET Framework et .NET Standard de Unity, avec .NET Standard 2.1 et .NET 4.x.
    - Intégrer Amazon GameLift dans un projet Unity
  - Le plugin autonome pour Unity 2021.3 LTS et 2022.3 LTS est un plugin complet avec les bibliothèques du SDK C# conçues pour Unity et des éléments d'interface graphique pour configurer et déployer les ressources Amazon pour l'hébergement. GameLift

    - Guide du GameLift plugin Amazon pour Unity pour le SDK de serveur 5.x
    - Référence GameLift du SDK Amazon Server pour C#
- Moteur Unreal
    - Le plugin C++ Server SDK pour Unreal est un plugin léger composé du code source C++ Unreal que vous pouvez intégrer dans des bibliothèques pour une utilisation avec les versions 4, 5 et 5.1 d'Unreal Engine.
        - Intégrer Amazon GameLift dans un projet Unreal Engine
        - Référence du SDK GameLift 5.x du serveur Amazon Unreal Engine
    - Le plugin autonome pour Unreal Engine 5.0, 5.1 et 5.2 est un plugin complet avec les bibliothèques et le SDK C++ pour Unreal Server SDK. AWS Le plugin est installé dans l'éditeur Unreal, avec des éléments d'interface utilisateur et des supports pour configurer et déployer les GameLift ressources Amazon pour l'hébergement.
        - Intégration de jeux avec le GameLift plugin Amazon pour Unreal Engine
        - Référence du SDK GameLift 5.x du serveur Amazon Unreal Engine

Systèmes d'exploitation de serveur de jeux

Utilisez le SDK Amazon GameLift Server pour créer des serveurs de jeu destinés à fonctionner sur les plateformes suivantes :

- Windows Server 2016
- Amazon Linux 2023
- Amazon Linux 2 (AL2)
- Windows Server 2012 (voir la GameLift FAQ Amazon pour Windows 2012)
- Amazon Linux (AL1) (voir la GameLift FAQ Amazon pour AL1)

Pour des services clients personnalisés

Créez des services clients personnalisés 64 bits à l'aide du AWS SDK avec l' GameLift API Amazon. Ce SDK permet aux services clients de gérer les sessions de jeu et d'associer les joueurs à des jeux hébergés sur Amazon GameLift. Pour commencer, téléchargez le AWS SDK. Pour plus d'informations sur l'utilisation du SDK avec Amazon GameLift, consultez le Amazon GameLift API Reference.

Pour serveurs en temps réel

Configurez et déployez des serveurs en temps réel pour héberger vos parties multijoueurs. Pour permettre à vos clients de jeu de se connecter à des serveurs en temps réel, utilisez le SDK Amazon GameLift Realtime Client. Les clients de jeu utilisent ce SDK pour échanger des messages avec un serveur en temps réel et avec d'autres clients de jeu qui se connectent au serveur. Pour commencer, téléchargez le SDK Amazon GameLift Realtime Client. Pour obtenir des informations de configuration, consultezIntégration d'un client de jeu pour les serveurs en temps réel.

Prise en charge de SDK

Le SDK du client en temps réel contient le code source pour les langues suivantes :

- C# (.NET)

Environnements de développement

Créez le SDK à partir des sources en fonction des besoins des systèmes d'exploitation de développement et moteurs de jeu pris en charge suivants :

- Systèmes d'exploitation : Windows, Linux, Android, iOS
- Moteurs de jeu — Unity, moteurs compatibles avec les bibliothèques C#

Systèmes d'exploitation de serveur de jeux

Vous pouvez déployer des serveurs en temps réel sur des ressources d'hébergement exécutées sur les plateformes suivantes :

- Amazon Linux
- Amazon Linux 2

Gérez les coûts d'hébergement de vos jeux

Votre AWS facture reflète les frais d'hébergement de vos jeux. Vous pouvez consulter les frais estimés pour le mois en cours et les frais finaux des mois précédents sur la console de facturation à l'adresse https://console.aws.amazon.com/billing/. Pour plus d'informations sur les outils et les ressources destinés à vous aider à gérer vos AWS coûts, consultez le guide de AWS Billing

l'utilisateur. Ce guide peut vous aider à évaluer votre consommation de ressources, à déterminer votre utilisation future et à déterminer vos besoins en matière de mise à l'échelle.

Tenez compte en particulier de ces conseils pour vous aider à gérer le coût des GameLift services Amazon.

Créez des alertes de facturation pour surveiller l'utilisation

Configurez une alerte d'utilisation du niveau AWS gratuit pour vous avertir lorsque votre utilisation approche ou dépasse les limites du niveau gratuit pour Amazon GameLift et les autresServices AWS. Vous pouvez configurer les alertes pour qu'elles prennent des mesures en fonction de vos niveaux d'utilisation. Par exemple, vous pouvez définir automatiquement votre budget à zéro lorsque vous atteignez la limite du niveau gratuit.

Vous pouvez également configurer des alertes CloudWatch de facturation Amazon pour recevoir des notifications lorsque l'utilisation atteint des seuils personnalisés.

Pour plus d'informations, consultez les rubriques suivantes dans le guide de AWS Billing l'utilisateur :

- Suivi de votre utilisation du niveau AWS gratuit
- Préférences relatives aux alertes de facturation

Suivez les coûts par GameLift flotte Amazon

Utilisez des balises de répartition des AWS coûts pour organiser et suivre les coûts d'hébergement de vos jeux en fonction des flottes GameLift Amazon EC2 et des autres ressources EC2. En étiquetant vos flottes, individuellement ou par groupes, vous pouvez créer des rapports de répartition des coûts qui catégorisent les coûts en fonction de l'étiquette attribuée. Vous pouvez utiliser ce type de rapport pour identifier la manière dont les flottes contribuent à vos coûts d'hébergement. Vous pouvez également utiliser les balises pour filtrer les vues dans AWS Cost Explorer.

Pour en savoir plus, consultez les rubriques suivantes :

- Utilisation des balises de répartition des AWS coûts, guide de AWS Billing l'utilisateur
- Analyse de vos AWS coûts avec Cost Explorer, guide de AWS Cost Management l'utilisateur

Réduisez à zéro la capacité inutilisée du parc

Les flottes peuvent continuer à encourir des coûts même lorsqu'elles ne sont pas utilisées pour héberger des sessions de jeu. Pour éviter d'encourir des frais inutiles, réduisez votre parc à zéro lorsqu'il n'est pas utilisé. Si vous utilisez le dimensionnement automatique, suspendez cette activité et définissez manuellement la capacité du parc.

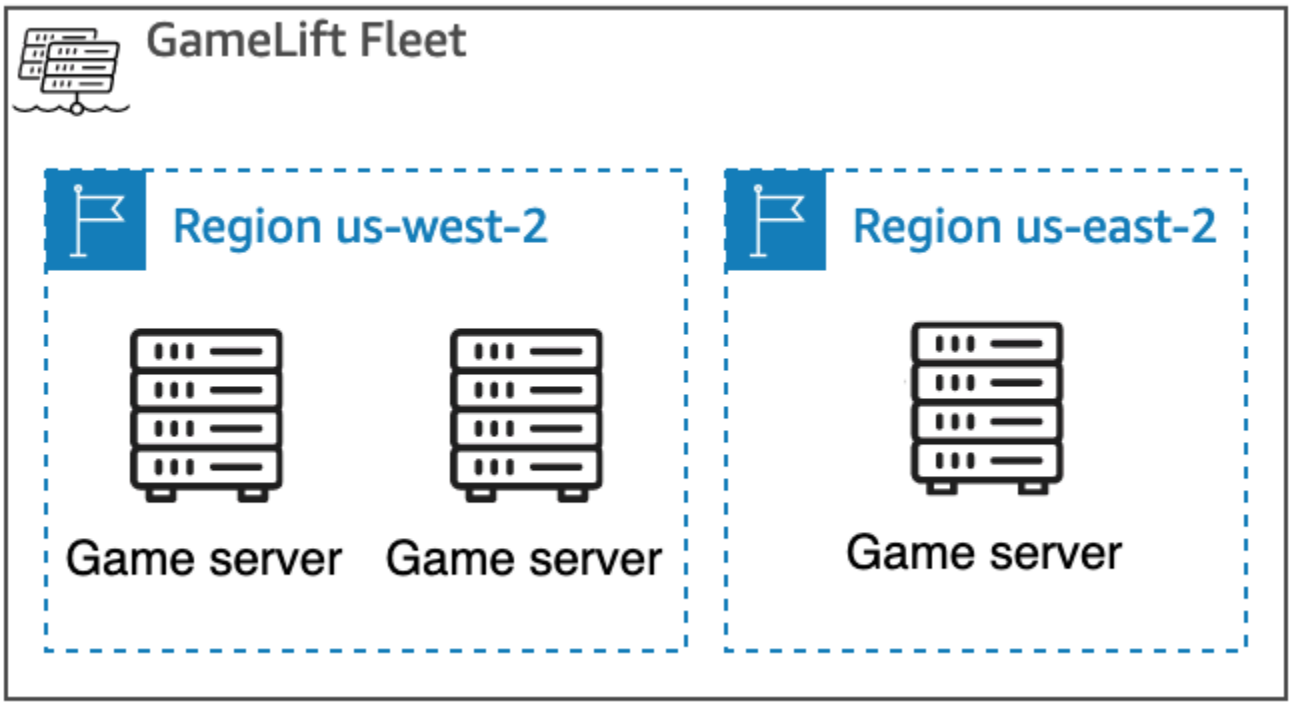
Sites GameLift d'hébergement Amazon

Amazon GameLift est disponible dans plusieurs Régions AWS zones locales. Pour une liste complète des sites, consultez la section GameLiftPoints de terminaison et quotas Amazon dans le Références générales AWS.

GameLift Hébergement Amazon

Lorsque vous créez une GameLift flotte Amazon, Amazon GameLift crée les ressources de la flotte dans votre flotte actuelleRégion AWS. Amazon GameLift appelle cette région la région d'origine de la flotte. Pour gérer une flotte, accédez-y depuis sa région d'origine.

Les flottes multisites déploient des instances sur d'autres sites en plus de la région d'origine de la flotte. Avec les flottes multisites, vous pouvez gérer la capacité de chaque site individuellement, et vous pouvez organiser les sessions de jeu par site. Les flottes multi-sites peuvent avoir des sites distants dans n'importe quelle région ou zone locale prise en charge par Amazon GameLift . Le schéma suivant illustre une flotte à sites multiples avec des ressources dans deux régions. Dans le schéma, la `us-west-2` région comprend deux serveurs de jeu, et la `us-east-2` région possède un serveur de jeu.



Si vous choisissez d'utiliser un parc multi-sites avec des instances situées dans des régions qui ne sont pas activées par défaut, vous devez activer ces régions dans votreCompte AWS. De plus, votre politique GameLift d'administrateur Amazon doit autoriser cette `ec2:DescribeRegions` action. Pour plus d'informations sur les régions qui ne sont pas activées par défaut et sur la manière de les activer, consultez la section Gestion Régions AWS dans le Références générales AWS. Pour un exemple de politique avec des régions qui ne sont pas activées par défaut, consultezExemples d'autorisations d'administrateur.

> **Important**
>
> Pour utiliser des régions qui ne sont pas activées par défaut, activez-les dans votreCompte AWS.
>
> - Les flottes dont les régions ne sont pas activées et que vous avez créées avant le 28 février 2022 ne sont pas affectées.
> - Pour créer de nouvelles flottes multi-sites ou pour mettre à jour les flottes multi-sites existantes, activez d'abord les régions que vous choisissez d'utiliser.

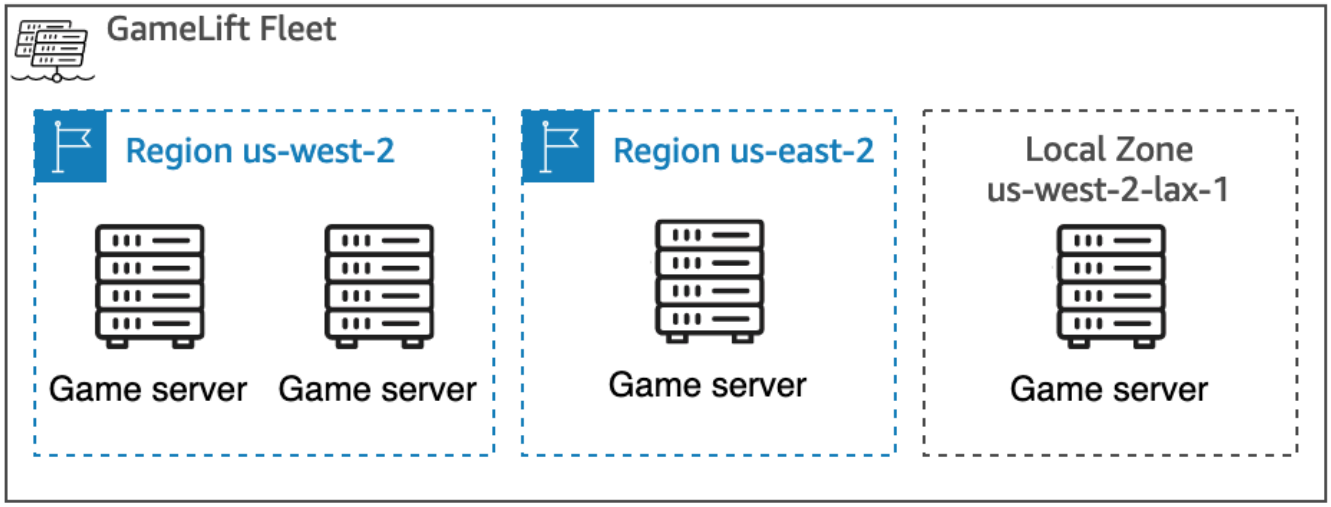
Pour le placement des sessions de jeu, vous pouvez créer des files d'attente pour les sessions de jeu dans n'importe quel endroit pris en GameLift charge par Amazon. Amazon GameLift place les sessions de jeu à partir de l'endroit où vous avez créé la file d'attente.

Zones locales

Une zone locale est une extension de la proximité géographique avec vos utilisateurs. Région AWS Les Zones Locales disposent de leurs propres connexions à Internet. Les zones locales sont également prises en charge AWS Direct Connect afin que les ressources créées dans une zone locale puissent desservir les utilisateurs locaux avec des communications à faible latence. Pour plus d'informations, consultez Local ZonesAWS.

Le code d'une zone locale est son code de région, suivi d'un identifiant indiquant son emplacement physique. Par exemple, la zone `us-west-2-lax-1` locale se trouve à Los Angeles. Pour obtenir la liste des Zones Locales disponibles, consultezLocal Zones disponibles.

Amazon GameLift héberge vos jeux dans chacun des lieux que vous choisissez pour votre flotte. Le schéma suivant représente une flotte composée de deux serveurs de jeu dans la `us-west-2` région, d'un serveur de jeu dans la `us-east-2` région et d'un serveur de jeu dans la zone `us-west-2-lax-1` locale.



Local Zones disponibles

Le tableau suivant répertorie les Zones Locales disponibles et leurs emplacements physiques.

Zone locale	Emplacement (zone métropolitaine)	
us-east-1-atl-1	Atlanta	
us-east-1-chi-1	Chicago	
us-east-1-dfw-1	Dallas	
us-east-1-iah-1	Houston	
us-east-1-mci-1	Kansas City	
us-west-2-den-1	Denver	
us-west-2-lax-1	Los Angeles	
us-west-2-phx-1	Phoenix	

Amazon GameLift Anywhere

Vous pouvez utiliser Amazon GameLift Anywhere pour créer des flottes avec votre propre matériel et gérer vos builds de jeux, vos scripts, vos serveurs de jeux et vos clients à l'aide d'Amazon GameLift. Amazon GameLift Anywhere est disponible dans toutes les régions prises en GameLift charge par Amazon. Pour plus d'informations sur la création d'une Anywhere flotte et le test de l'intégration de votre serveur de jeu, consultez Créez une GameLift Anywhere flotte Amazon etTestez votre intégration à l'aide des GameLift Anywhere flottes Amazon.

Avec Amazon, GameLift Anywhere vous créez des emplacements personnalisés qui représentent l'emplacement physique du matériel que vous utilisez pour héberger vos serveurs de jeu GameLift intégrés Amazon.

Amazon GameLift FlexMatch

En effet FlexMatch, vous pouvez organiser des sessions de jeu générées par des matchs dans n'importe quel endroit pris en charge par Amazon GameLift. L'activité de matchmaking réelle a lieu Région AWS là où vous avez choisi de créer vos ressources de matchmaking. Amazon GameLift achemine les demandes de correspondance vers le système de jumelage et les traite à cet endroit.

Pour plus d'informations sur Amazon GameLift FlexMatch, consultez Qu'est-ce qu'Amazon GameLift FlexMatch ?

Régions AWSqui soutiennent les FlexMatch ressources

Amazon GameLift en Chine

Lorsque vous utilisez Amazon GameLift pour des ressources dans la région de Chine (Pékin), gérée par Sinnet, ou dans la région de Chine (Ningxia), gérée par NWCD, vous devez disposer d'un compte distinct AWS (Chine). Notez que certaines fonctionnalités ne sont pas disponibles dans les régions de Chine. Pour plus d'informations sur l'utilisation d'Amazon GameLift dans ces régions, consultez les ressources suivantes :

- Amazon Web Services en Chine
- Amazon GameLift (Commencer à utiliser Amazon Web Services en Chine)

Débuter avec Amazon GameLift

Nous vous recommandons d'essayer les exemples suivants avant d'utiliser Amazon GameLift pour votre propre jeu. L'exemple de serveur de jeu personnalisé vous permet de vous familiariser avec l'hébergement de jeux sur la GameLift console Amazon. L'exemple de serveurs en temps réel vous montre comment préparer un jeu pour l'hébergement à l'aide de serveurs en temps réel.

Pour commencer à utiliser Amazon GameLift pour votre propre jeu, consultezFeuille de route d'hébergement GameLift géré par Amazon.

Exemple de serveur de jeu personnalisé

Cet exemple montre un jeu personnalisé en direct sur AmazonGameLift. L'exemple vous guide à travers les étapes suivantes :

- Création d'un exemple de build de jeu.
- Création d'une flotte pour faire fonctionner le serveur de jeu.
- Connexion au serveur de jeu à partir du client de jeu d'exemple.
- Examiner les statistiques de la flotte et des sessions de jeu.

Après ces étapes, vous pouvez démarrer plusieurs clients de jeu et jouer au jeu pour générer des données d'hébergement. Vous pouvez ensuite explorer la GameLift console Amazon pour consulter vos ressources d'hébergement, suivre les indicateurs et expérimenter des moyens d'adapter la capacité d'hébergement.

Pour commencer, connectez-vous à la GameLiftconsole Amazon.

Exemple de jeu Realtime Servers

Cet exemple est un jeu multijoueur complet nommé Mega Frog Race, avec le code source inclus. L'exemple montre comment intégrer votre client de jeu à des serveurs en temps réel. Vous pouvez également utiliser cet exemple de jeu comme point de départ pour expérimenter d'autres GameLift fonctionnalités d'Amazon telles queFlexMatch.

Pour un didacticiel pratique, consultez la section Création de serveurs pour les jeux mobiles multijoueurs en quelques lignes JavaScript sur le blog AWS for Games.

Pour le code source de Mega Frog Race, consultez le GitHubréférentiel.

Le code source comprend les parties suivantes :

- Client de jeu : code source pour le client de jeu C++, créé dans Unity. Le client du jeu obtient les informations de connexion à la session de jeu, se connecte au serveur et échange des mises à jour avec les autres joueurs.
- Service principal : code source d'une AWS Lambda fonction qui gère les appels d'API directs vers AmazonGameLift.
- Script en temps réel : fichier de script source qui configure un parc de serveurs en temps réel pour le jeu. Ce script inclut la configuration minimale requise pour que les serveurs en temps réel puissent communiquer avec Amazon GameLift et héberger des jeux.

Feuille de route d'hébergement GameLift géré par Amazon

Cette rubrique vous aide à choisir parmi les différentes options GameLift d'hébergement Amazon pour votre jeu multijoueur basé sur des sessions. Les autres rubriques de cette section vous expliquent comment utiliser Amazon GameLift pour votre hébergement géré.

Avant de commencer à préparer le lancement de votre jeu en production, remplissez le questionnaire de lancement pour commencer à travailler avec l'GameLiftéquipe Amazon.

Rubriques

Choisissez une option d'hébergement

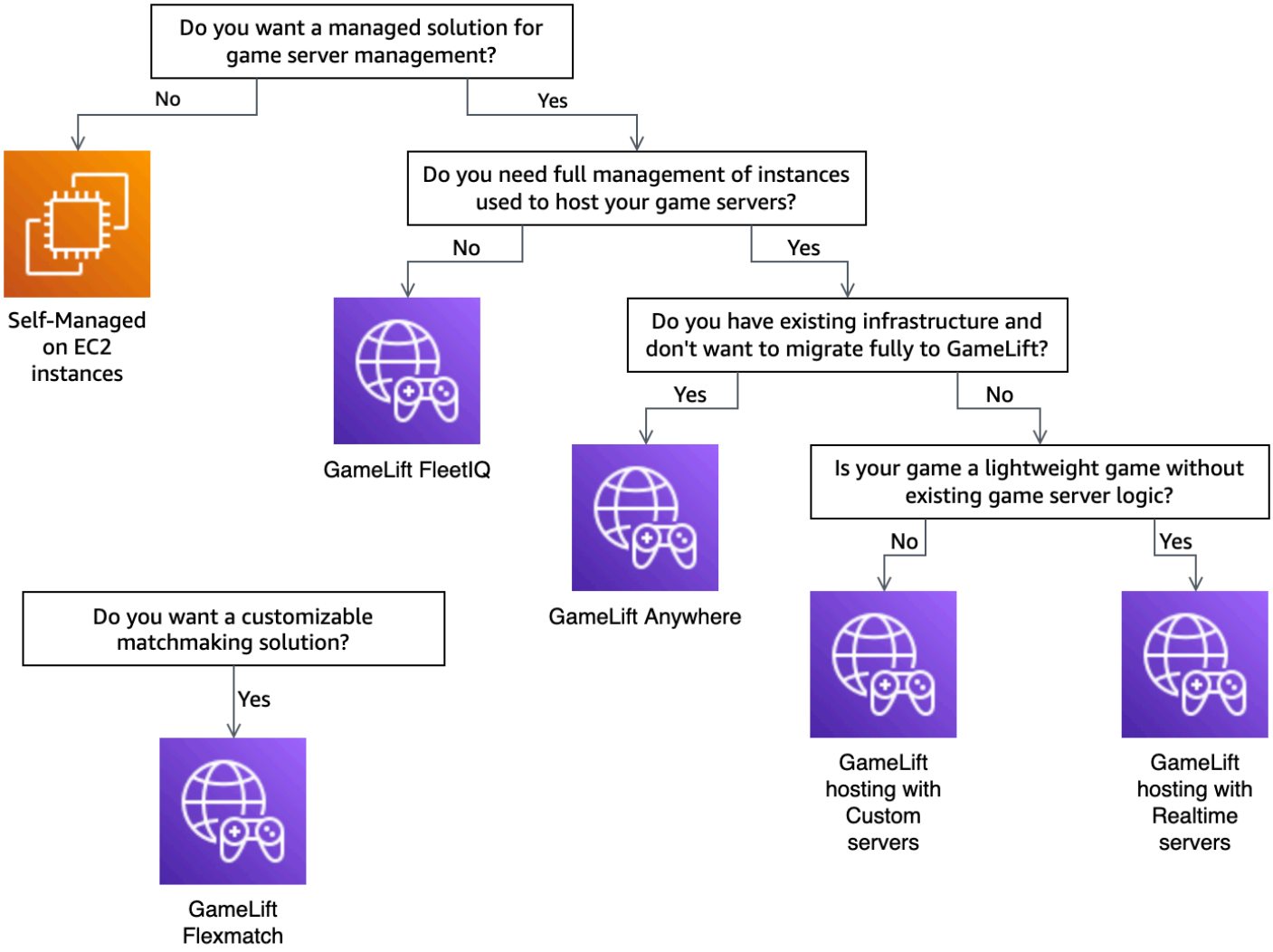
L'organigramme suivant pose des questions qui vous guideront vers la GameLift solution Amazon adaptée à votre cas d'utilisation.

1. Vous souhaitez une solution gérée pour la gestion des serveurs de jeux ?
   - Oui, passez à l'étape 2.
   - Non. Envisagez d'utiliser des serveurs de jeu autogérés sur des instances Amazon EC2.
2. Avez-vous besoin d'un contrôle total sur les instances hébergeant vos serveurs de jeu ?
   - Oui, pensez à Amazon GameLift FleetIQ.
   - Non — Passez à l'étape 3.
3. Disposez-vous d'une infrastructure existante que vous souhaitez utiliser avec Amazon GameLift ?
   - Oui, pensez à Amazon GameLiftAnywhere.

- Non — Passez à l'étape 4.

4. Votre jeu est-il léger sans la logique du serveur de jeu existante ?

   - Oui, pensez aux serveurs en temps réel.
   - Non, pensez à des serveurs personnalisés.



Voici quelques informations supplémentaires sur certaines des options d'GameLifthébergement Amazon mentionnées dans l'organigramme :

Amazon GameLift Anywhere

Utilisez Amazon GameLift Anywhere pour héberger vos jeux sur votre propre matériel en profitant des outils de GameLift gestion Amazon. Vous pouvez également utiliser des Anywhere flottes

pour tester vos serveurs de jeu de manière itérative. Pour plus d'informations, veuillez consulter Créez une GameLift Anywhere flotte Amazon.

Amazon géré GameLift

Il existe deux options pour l'GameLiftthébergement Amazon géré :

Serveurs personnalisés — Amazon GameLift héberge votre serveur personnalisé qui exécute le binaire de votre serveur de jeu.

Serveurs en temps réel : Amazon GameLift héberge votre serveur de jeu léger.

Amazon GameLift FleetIQ

Dans l'organigramme, une migration ascendante fait référence à une migration au cours de laquelle vous ne pouvez pas apporter de modifications à l'architecture du jeu. L'utilisation d'Amazon GameLift FleetIQ nécessite moins de modifications à votre déploiement existant et fournit des GameLift outils Amazon pour la gestion de flotte. Pour plus d'informations, consultez le guide du développeur Amazon GameLift FleetIQ.

Si vous décidez d'utiliser Amazon GameLift Anywhere ou Amazon géréGameLift, continuez à le fairePréparez votre jeu pour Amazon GameLift.

Préparez votre jeu pour Amazon GameLift

Cette rubrique décrit les étapes à suivre pour préparer votre jeu multijoueur en vue de son intégration à l'GameLiftthébergement Amazon géré. Pour préparer votre jeu, vous devez activer la communication entre celui-ci et AmazonGameLift.

Préparez votre serveur de jeu personnalisé

Pour démarrer et arrêter des sessions de jeu et pour effectuer d'autres tâches, un serveur de jeu doit être en mesure d'GameLiftinformer Amazon de son état. Pour activer la communication avec AmazonGameLift, ajoutez du code à votre projet de serveur de jeu. Pour plus d'informations, veuillez consulter Intégrez des jeux à des serveurs de jeux personnalisés.

1. Préparez votre serveur de jeu personnalisé pour l'hébergement sur AmazonGameLift.

   - Procurez-vous le SDK Amazon GameLift Server et créez-le pour votre langage de programmation et votre moteur de jeu préférés.

- Ajoutez du code à votre projet de serveur de jeu pour activer la communication avec AmazonGameLift.

2. Préparez votre client de jeu pour qu'il se connecte aux sessions de jeu GameLift hébergées par Amazon.

   - Ajoutez le AWS SDK à votre service principal et à votre projet de client de jeu. Pour plus d'informations, consultez la section Télécharger les GameLift kits SDK Amazon pour les services client.
   - Ajoutez des fonctionnalités permettant de récupérer des informations sur les sessions de jeu, d'en créer de nouvelles et de réserver de l'espace aux joueurs lors d'une session de jeu.
   - (Facultatif) À utiliser FlexMatch pour le matchmaking des joueurs. Pour plus d'informations, consultez la section FlexMatchIntégration avec l'GameLifthébergement Amazon.

Préparez votre serveur en temps réel

Amazon GameLift Realtime Servers fournit une solution de serveur légère que vous pouvez configurer en fonction de votre jeu. Un serveur en temps réel offre les mêmes avantages qu'Amazon GameLift offre aux serveurs de jeux, mais avec une personnalisation réduite des serveurs de jeu.

Créez un script en temps réel pour l'hébergement sur AmazonGameLift.

Les scripts en temps réel contiennent la configuration de votre serveur et une logique de jeu personnalisée en option. Les serveurs en temps réel sont conçus pour démarrer et arrêter des sessions de jeu, accepter les connexions des joueurs et gérer les communications avec Amazon GameLift et entre les joueurs d'un jeu. Il existe également des outils qui vous permettent d'ajouter une logique de serveur personnalisée à votre jeu. Les serveurs en temps réel utilisent Node.js et. JavaScript Pour plus d'informations, consultez Création d'un script en temps réel et Testez votre intégration avec Amazon GameLift.

Testez votre intégration avec Amazon GameLift

Amazon GameLift prend en charge l'itération rapide lors du test de vos serveurs de jeu. Cette rubrique vous guide à travers les différents types de tests disponibles.

Serveurs de jeux personnalisés

Utilisez Amazon GameLift pour intégrer du matériel n'importe où dans votre environnement à votre architecture d'hébergement de GameLift jeux Amazon. Amazon GameLift Anywhere enregistre votre

matériel auprès d'Amazon GameLift dans une Anywhere flotte, afin que vous puissiez le tester à l'aide de votre propre ordinateur de développement local. Pour plus d'informations sur les tests avec Amazon GameLiftAnywhere, consultezTestez votre intégration à l'aide des GameLift Anywhere flottes Amazon. Pour plus d'informations sur l'utilisation d'Amazon GameLift Anywhere pour héberger vos jeux avec des solutions sur site, consultezChoisir les ressources GameLift informatiques d'Amazon.

Serveurs en temps réel

Avec les serveurs en temps réel, vous pouvez mettre à jour vos scripts à tout moment. Lorsque vous mettez à jour un script en temps réel, Amazon GameLift distribue la nouvelle version à vos ressources d'hébergement en quelques minutes. Une fois qu'Amazon GameLift a déployé le nouveau script, toutes les nouvelles sessions de jeu utilisent la nouvelle version du script. Une fois qu'Amazon GameLift a déployé le nouveau script, vous pouvez commencer les tests immédiatement. Pour plus d'informations sur les serveurs en temps réel, voir, Intégration de jeux aux serveurs Amazon GameLift Realtime

Planifiez et déployez vos GameLift ressources Amazon

Suivez les conseils suivants pour vous aider à planifier le déploiement de vos GameLift ressources Amazon à l'échelle mondiale. Pour en savoir plus sur les sites où vous pouvez héberger vos jeux sur AmazonGameLift, consultezSites GameLift d'hébergement Amazon.

Pour déployer vos GameLift ressources Amazon, effectuez les tâches suivantes :

- Empaquetez et chargez votre serveur de jeu sur Amazon GameLift ou sur votre matériel. Lorsque vous importez votre serveur sur AmazonGameLift, vous le chargez uniquement sur le site d'accueil Région AWS de votre flotte. Amazon distribue GameLift automatiquement le serveur aux autres sites de la flotte. Pour plus d'informations, veuillez consulter Téléchargement de versions et de scripts sur Amazon GameLift.
- Concevez et déployez une GameLift flotte Amazon pour votre jeu. Déterminez le type de ressources informatiques à utiliser, les emplacements vers lesquels déployer, s'il convient d'utiliser des files d'attente et d'autres options. Pour plus d'informations, veuillez consulter Guide GameLift de conception de flotte Amazon.
- Créez des files d'attente pour gérer l'emplacement des nouvelles sessions de jeu et les stratégies des instances Spot. Pour plus d'informations, veuillez consulter Conception d'une file d'attente de sessions de jeu.

- Utilisez la mise à l'échelle automatique pour gérer la capacité d'hébergement de votre flotte en fonction de la demande attendue des joueurs. Pour plus d'informations, veuillez consulter Élargir la capacité GameLift d'hébergement d'Amazon.
- Utilisez les règles de FlexMatch matchmaking pour votre jeu. Pour plus d'informations, consultez la section FlexMatchIntégration avec l'GameLifthébergement Amazon.

Déployez automatiquement vos GameLift ressources Amazon

Pour rationaliser le déploiement mondial de vos GameLift ressources Amazon, nous vous recommandons d'utiliser l'infrastructure en tant que code (IaC) pour définir les ressources. Étant donné qu'Amazon GameLift prend en charge les AWS CloudFormation modèles, vous pouvez définir des paramètres dans les modèles pour toutes les configurations spécifiques au déploiement.

Pour gérer le déploiement de vos AWS CloudFormation stacks, nous vous recommandons également d'utiliser des outils et des services d'intégration continue et de livraison continue (CI/CD) tels que. AWS CodePipeline Ils vous aident à effectuer un déploiement automatique ou avec approbation chaque fois que vous créez un fichier binaire pour un serveur de jeu.

Voici quelques étapes courantes du déploiement des GameLift ressources Amazon pour une nouvelle version de serveur de jeu que vous pouvez automatiser à l'aide d'un outil ou d'un service CI/CD :

- Création et test du binaire de votre serveur de jeu.
- Chargez le fichier binaire sur Amazon GameLift ou sur votre matériel.
- Déploiement de nouvelles flottes dans la nouvelle version.
- Après avoir déployé les nouvelles flottes, supprimez les flottes de la version précédente de votre GameLift file d'attente Amazon et remplacez-les par les nouvelles flottes.
- Une fois que les flottes de la version précédente ont terminé toutes les sessions de jeu avec succès, les AWS CloudFormation piles de ces flottes sont supprimées.

Vous pouvez également utiliser le AWS Cloud Development Kit (AWS CDK) pour définir vos GameLift ressources Amazon. Pour en savoir plus sur le AWS CDK, veuillez consulter le Guide du développeur AWS Cloud Development Kit (AWS CDK).

Concevez le service client de votre jeu

Nous vous recommandons de mettre en place un service client de jeu qui authentifie vos joueurs et communique avec l'API AmazonGameLift. En mettant en œuvre un service client de jeu personnalisé, vous pouvez :

- Personnalisez l'authentification de vos joueurs.
- Contrôlez la façon dont Amazon GameLift fait correspondre et démarre les sessions de jeu.
- Utilisez votre base de données de joueurs pour les attributs des joueurs tels que le classement des compétences pour le matchmaking au lieu de faire confiance au client.

L'utilisation d'un service client de jeu réduit également les risques de sécurité liés aux clients de jeu qui interagissent directement avec votre GameLift API Amazon.

Authentifier vos joueurs

Vous pouvez utiliser Amazon Cognito et les identifiants de session des joueurs pour authentifier vos clients de jeu. Pour gérer le cycle de vie et les propriétés de vos identités de joueurs, utilisez les groupes d'utilisateurs Amazon Cognito.

Si vous préférez, créez une solution d'identité personnalisée et hébergez-la surAWS. Vous pouvez également utiliser les autorisateurs Lambda pour une logique d'autorisation personnalisée avec API Gateway.

Ressources supplémentaires :

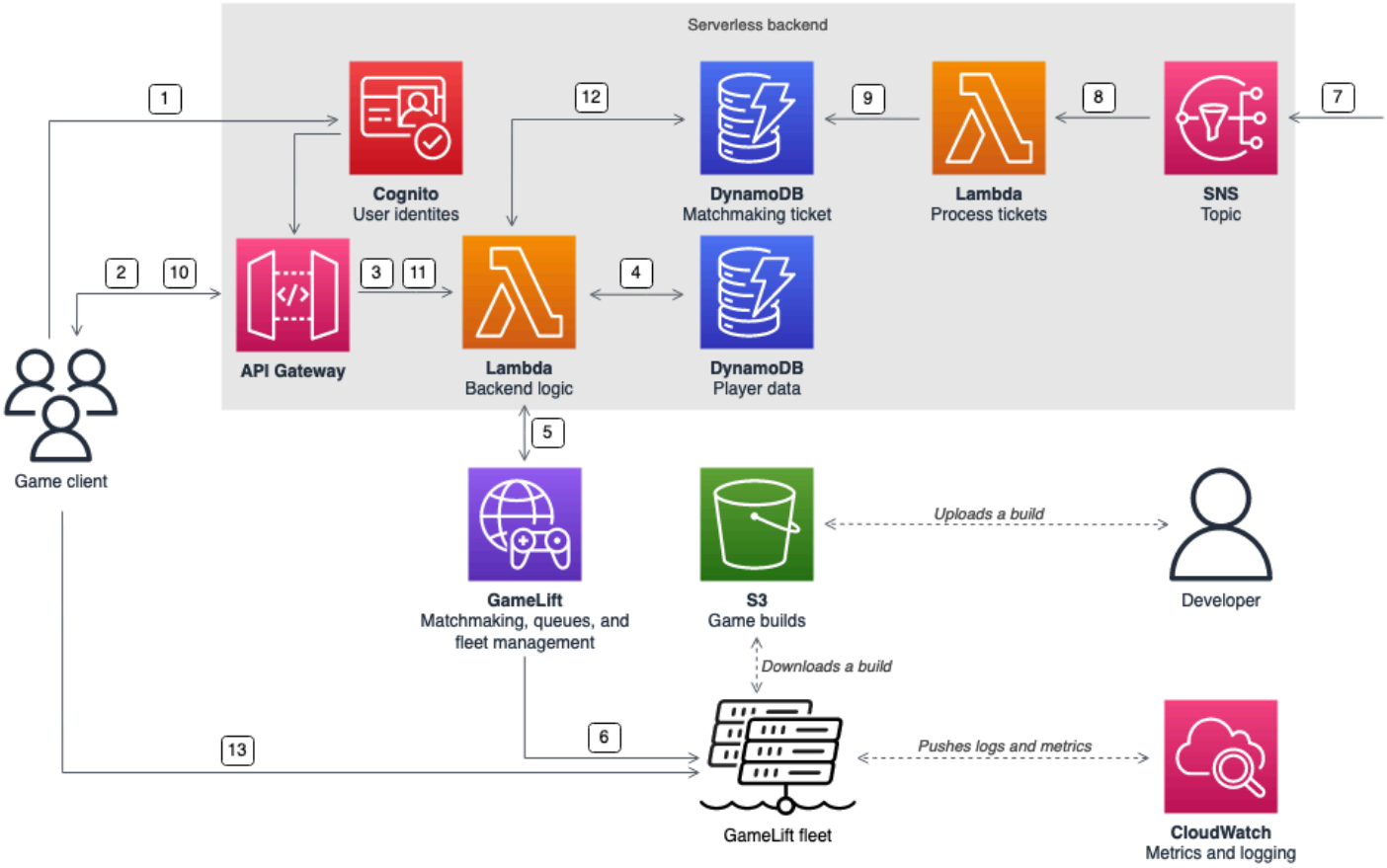
- Utilisation de pools d'identités (identités fédérées) (Amazon Cognito Developer Guide)
- Premiers pas avec les groupes d'utilisateurs (Amazon Cognito Developer Guide)
- Comment configurer l'authentification des joueurs avec Amazon Cognito (AWSpour Games Blog)

Serveurs de session de jeu autonomes avec backend sans serveur

À l'aide d'une architecture de service client sans serveur, le backend peut consulter l'état des tickets de matchmaking à partir d'une base de données hautement évolutive au lieu d'accéder directement à l'API AmazonGameLift.

Le schéma suivant montre un backend sans serveur intégré Services AWS qui associe les joueurs à des jeux exécutés sur des flottes AmazonGameLift. La liste suivante fournit une description

de chaque légende numérotée du diagramme. Pour essayer cet exemple, consultez la section Hébergement de jeux basé sur des sessions multijoueurs sur on. AWS GitHub



1. Le client du jeu demande une identité utilisateur Amazon Cognito à partir d'un pool d'identités Amazon Cognito.
2. Le client du jeu reçoit des identifiants d'accès temporaires et demande une session de jeu via une API Amazon API Gateway.
3. API Gateway invoque n'importe quelle AWS Lambda fonction.
4. La fonction Lambda demande les données des joueurs à partir d'une table Amazon DynamoDB NoSQL. La fonction fournit l'identité Amazon Cognito dans les données contextuelles de la demande.
5. La fonction Lambda demande une correspondance via Amazon GameLift FlexMatch matchmaking.
6. FlexMatchmet en relation un groupe de joueurs avec une latence appropriée, puis demande le placement d'une session de jeu via une GameLift file d'attente Amazon. La file contient des flottes comportant un ou plusieurs Région AWS emplacements.

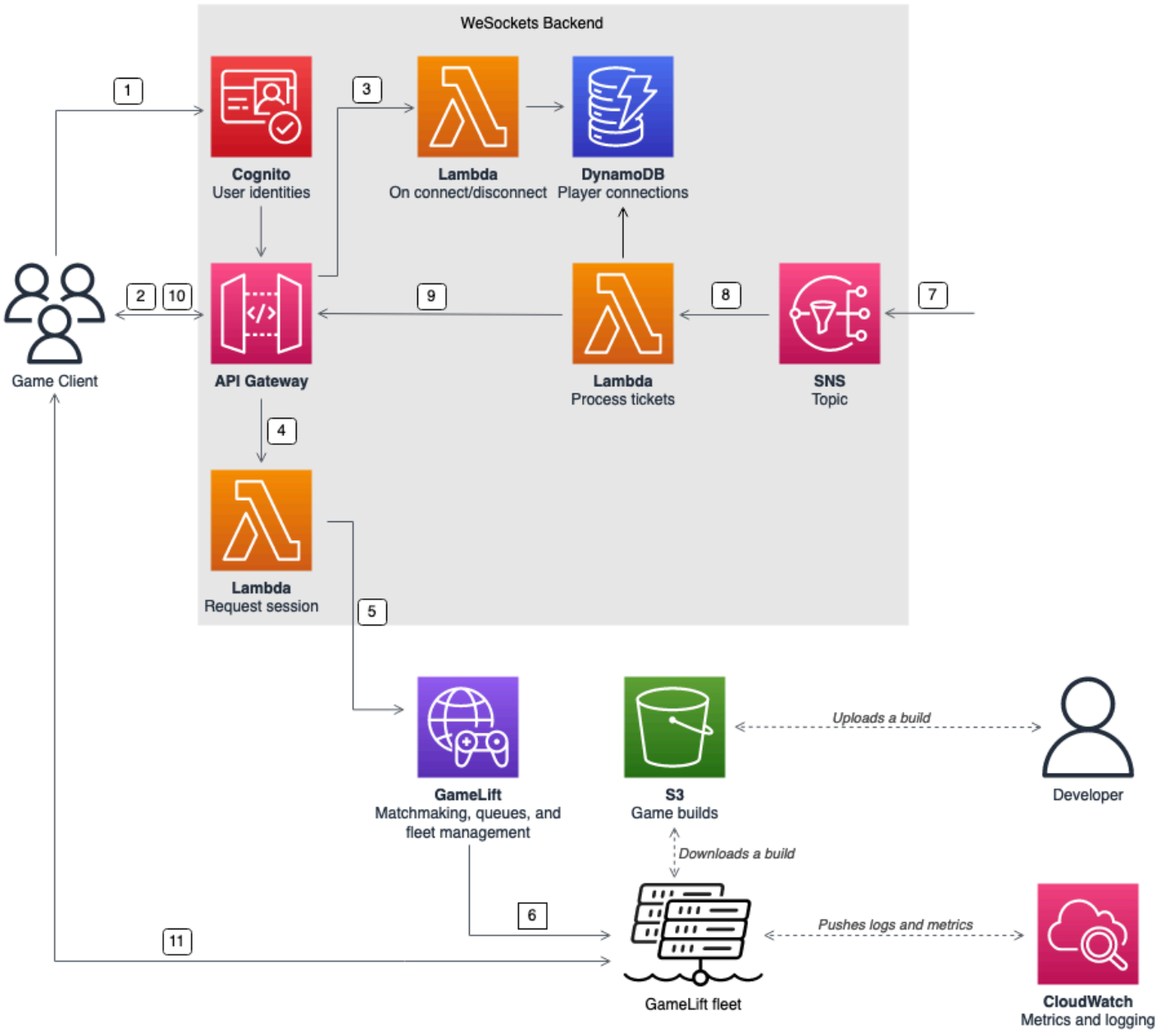
7. Une fois qu'Amazon GameLift a placé la session sur l'un des sites de la flotte, Amazon GameLift envoie une notification d'événement à une rubrique Amazon Simple Notification Service (Amazon SNS).
8. Une fonction Lambda reçoit l'événement Amazon SNS et le traite.
9. Si le ticket de matchmaking est un `MatchmakingSucceeded` événement, la fonction Lambda écrit le résultat, ainsi que le port et l'adresse IP du serveur de jeu, dans une table DynamoDB.
10. Le client du jeu envoie une demande signée à API Gateway pour consulter l'état du ticket de matchmaking à un intervalle spécifique.
11. API Gateway utilise une fonction Lambda qui vérifie l'état du ticket de matchmaking.
12. La fonction Lambda vérifie la table DynamoDB pour voir si le ticket est un succès. En cas de succès, la fonction renvoie le port et l'adresse IP du serveur de jeu, ainsi que l'identifiant de session du joueur, au client. Si le ticket échoue, la fonction envoie une réponse confirmant que le match n'est pas encore prêt.
13. Le client du jeu se connecte au serveur de jeu via TCP ou UDP en utilisant le port et l'adresse IP fournis par le service principal. Le client du jeu envoie ensuite l'identifiant de session du joueur au serveur de jeu, qui valide ensuite l'identifiant à l'aide du SDK Amazon GameLift Server.

Serveurs de session de jeu autonomes avec un backend WebSocket basé

À l'aide d'une architecture WebSocket basée sur Amazon API Gateway, vous pouvez effectuer des demandes de matchmaking WebSockets et envoyer des notifications push pour la fin du matchmaking à l'aide de messages initiés par le serveur. Cette architecture améliore les performances en établissant une communication bidirectionnelle entre le client et le serveur.

Pour plus d'informations sur l'utilisation des API WebSock API Gateway, consultez la section Utilisation des WebSocket API.

Le schéma suivant montre une architecture de backend WebSocket basée sur API Gateway et d'autres outils Services AWS pour associer les joueurs à des jeux exécutés sur des GameLift flottes Amazon. La liste suivante fournit une description de chaque légende numérotée du diagramme.



1. Le client du jeu demande une identité utilisateur Amazon Cognito à partir d'un pool d'identités Amazon Cognito.
2. Le client du jeu signe une WebSocket connexion à une API API Gateway à l'aide des informations d'identification Amazon Cognito.
3. API Gateway appelle une AWS Lambda fonction sur la connexion. La fonction enregistre les informations de connexion dans une table Amazon DynamoDB.
4. Le client du jeu envoie un message à une fonction Lambda, via l'API API Gateway via la WebSocket connexion, pour demander une session.

5. Une fonction Lambda reçoit le message puis demande une correspondance via Amazon GameLift FlexMatch matchmaking.
6. Après un FlexMatch match avec un groupe de joueurs, FlexMatch demande le placement d'une session de jeu via une GameLift file d'attente Amazon.
7. Une fois qu'Amazon GameLift a placé la session sur l'un des sites de la flotte, Amazon GameLift envoie une notification d'événement à une rubrique Amazon Simple Notification Service (Amazon SNS).
8. Une fonction Lambda reçoit l'événement Amazon SNS et le traite.
9. Si le ticket de matchmaking est un `MatchmakingSucceeded` événement, la fonction Lambda demande à DynamoDB la connexion correcte du joueur. La fonction envoie ensuite un message au client du jeu via l'API API Gateway via la WebSocket connexion. Dans cette architecture, le client du jeu ne vérifie pas activement l'état du matchmaking.
10. Le client du jeu reçoit le port et l'adresse IP du serveur de jeu, ainsi que l'identifiant de session du joueur, via la WebSocket connexion.
11. Le client du jeu se connecte au serveur de jeu via TCP ou UDP à l'aide du port et de l'adresse IP fournis par le service principal. Le client du jeu envoie également l'identifiant de session du joueur au serveur de jeu, qui valide ensuite l'identifiant à l'aide du SDK Amazon GameLift Server.

Configuration des métriques et de la journalisation pour Amazon GameLift

Vous pouvez utiliser les données collectées à partir de vos serveurs et ressources de GameLift jeu Amazon pour identifier les anomalies. Vous pouvez également utiliser des mesures pour améliorer les performances.

Les principaux domaines à observer pour Amazon GameLift sont les suivants :

- Métriques GameLift de service Amazon — Amazon GameLift fournit des CloudWatch statistiques Amazon sur vos ressources, notamment les serveurs de jeux, les flottes, les files d'attente et. FlexMatch Vous pouvez trouver ces statistiques dans la GameLift console Amazon et dans la CloudWatch console. Pour plus d'informations sur GameLift les métriques Amazon dansCloudWatch, consultezSurveillez Amazon GameLift avec Amazon CloudWatch.
- Statistiques du serveur de jeu : Amazon ne GameLift peut pas accéder aux statistiques de votre serveur de jeu. Toutefois, vous pouvez envoyer des statistiques personnalisées CloudWatch directement depuis votre serveur de jeu à l'aide de l'CloudWatchagent. Vous pouvez également

utiliser le rôle de flotte AWS Identity and Access Management (IAM) et le AWS SDK pour envoyer des métriques directement à. CloudWatch Pour un exemple de configuration des métriques, voir Hébergement de jeux basé sur des sessions multijoueurs sur activé. AWS GitHub Ce référentiel inclut un exemple de configuration d'CloudWatchagent et de code pour un client C# StatsD.

- Journaux du serveur de jeu : pour configurer les fichiers journaux de votre serveur de jeu sur le serveur de jeu, utilisez la configuration GameLift du SDK Amazon Server. Vous pouvez également utiliser Amazon CloudWatch Logs comme solution de gestion des journaux en temps réel et configurer les journaux avec l'CloudWatchagent. Pour plus d'informations, veuillez consulter Enregistrement des messages du serveur dans Amazon GameLift.

Listes de contrôle pour le lancement du jeu

Vous pouvez utiliser ces listes de contrôle pour valider les phases de déploiement de votre jeu. Dans les listes de contrôle, les éléments marqués [Critique] sont essentiels pour le lancement de votre production.

Rubriques

Intégration

Utilisez la liste de contrôle suivante pour suivre les éléments nécessaires à l'intégration de votre jeu pour l'hébergement AmazonGameLift. Les éléments marqués [Critique] sont essentiels pour le lancement de votre production.

- [Critique] Remplissez le questionnaire GameLift d'intégration Amazon dans la GameLiftconsole Amazon.
- [Critique] Concevez et implémentez un service principal permettant aux clients de jeu d'interagir avec vos serveurs de jeu.
- [Critique] Créez des rôles AWS Identity and Access Management (IAM) que vous fournissez aux instances de GameLift serveur Amazon pour accéder à d'autres AWS ressources.

- [Critique] Concevez et implémentez le basculement vers d'autres formulaires FlexMatch et Régions AWS files d'attente.
- Planifiez le déploiement des flottes vers vos destinations cibles en tenant compte de la file d'attente et de la structure de la flotte de votre jeu.
- Automatisez votre déploiement à l'aide de l'infrastructure sous forme de code (IaC) avec AWS CloudFormation et leAWS Cloud Development Kit (AWS CDK).
- Collectez des journaux et des analyses à l'aide d'Amazon CloudWatch et d'Amazon Simple Storage Service (Amazon S3).

Test

Utilisez la liste de contrôle suivante pour suivre les éléments de test lors du développement de votre jeu avec l'GameLifthébergement Amazon. Les éléments marqués [Critique] sont essentiels pour le lancement de votre production.

- [Critique] Complétez le questionnaire de lancement et soumettez-le à l'équipe de GameLift lancement d'Amazon. Vous pouvez trouver le questionnaire de lancement dans la GameLiftconsole Amazon.
- [Critique] Demandez une augmentation des quotas de GameLift service Amazon et d'autres Service AWS quotas afin que votre environnement réel puisse s'adapter aux besoins de production.
- [Critique] Vérifiez que les ports ouverts sur les flottes actives correspondent à la plage de ports que vos serveurs peuvent utiliser.
- [Critique] Fermez le port RDP 3389 et le port SSH 22.
- Élaborez un plan pour la DevOps gestion de votre jeu. Si vous utilisez Amazon CloudWatch Logs ou des métriques CloudWatch personnalisées d'Amazon, définissez des alarmes en cas de problèmes graves ou critiques sur le parc de serveurs. Simulez des défaillances et testez les runbooks.
- Vérifiez que le nombre de serveurs exécutés sur une instance à pleine utilisation correspond aux capacités du type d'instance de serveur.
- Ajustez votre politique de dimensionnement de manière à être plus prudente au début et à fournir une capacité inutilisée supérieure à ce dont vous pensez avoir besoin. Vous pouvez optimiser les coûts ultérieurement. Envisagez l'utilisation d'une politique de dimensionnement basée sur des objectifs avec 20 % de capacité inactive.

- Utilisez les règles de FlexMatch latence pour faire correspondre des joueurs géographiquement proches des mêmes joueursRégion AWS. Testez son comportement en cas de charge à l'aide des données de latence synthétiques de votre client de test de charge.
- Testez la charge de votre infrastructure d'authentification des joueurs et de session de jeu pour voir si elle s'adapte efficacement à la demande.
- Vérifiez qu'un serveur qui fonctionne depuis plusieurs jours peut toujours accepter des connexions.
- Élevez le niveau de votre AWS Support plan à Business ou Enterprise afin de AWS pouvoir répondre à vos besoins en cas de problème ou de panne.

Lancer

Utilisez la liste de contrôle suivante pour suivre les éléments de lancement de votre jeu hébergé sur AmazonGameLift. Les éléments marqués [Critique] sont essentiels pour le lancement de votre production.

- [Critique] Définissez la politique de protection de la flotte sur une protection complète pour toutes les flottes actives afin que la réduction de la taille n'interrompe pas les sessions de jeu actives.
- [Critique] Définissez des tailles maximales de flotte suffisamment élevées pour répondre au minimum aux pics de demande prévus. Nous vous recommandons de doubler votre taille maximale en cas de demande imprévue.
- Encouragez l'ensemble de votre équipe de développement à participer à l'événement de lancement et surveillez le lancement de votre jeu dans une salle de lancement.
- Surveillez la latence et l'expérience des joueurs.

Après le lancement

Utilisez la liste de contrôle suivante pour suivre les articles publiés après le lancement de votre jeu hébergé sur Amazon. GameLift

- Réglez les règles de dimensionnement pour minimiser la capacité inutilisée.
- Modifiez FlexMatch les règles ou ajoutez des emplacements supplémentaires en fonction de vos besoins en matière de latence.
- Optimisez l'exécutable du serveur, car l'efficacité de ses performances influe directement sur les coûts de la flotte. Pour exécuter davantage de sessions de jeu avec la même infrastructure, augmentez le nombre de processus serveur par instance.

- Utilisez vos données d'analyse pour favoriser le développement continu, améliorer l'expérience des joueurs et la longévité des jeux, et optimiser la monétisation.

Préparation de jeux pour Amazon GameLift

Pour préparer votre jeu multijoueur en vue de son hébergement sur Amazon GameLift, configurez la communication entre votre jeu et Amazon GameLift. Les rubriques de cette section fournissent une aide détaillée pour intégrer votre jeu à Amazon GameLift, aux serveurs de jeu personnalisés et aux serveurs en temps réel, et pour ajouter le matchmaking avec FlexMatch.

Rubriques

Intégrez des jeux à des serveurs de jeux personnalisés

Amazon GameLift fournit un ensemble complet d'outils pour préparer vos jeux multijoueurs et des serveurs de jeux personnalisés à exécuter sur AmazonGameLift. Les GameLift kits SDK Amazon contiennent les bibliothèques nécessaires aux clients et aux serveurs de jeux pour communiquer avec AmazonGameLift. Pour plus d'informations sur les kits SDK et pour savoir où les obtenir, consultezAssistance au développement avec Amazon GameLift.

Les rubriques de cette section contiennent des instructions détaillées sur la façon d'ajouter des GameLift fonctionnalités Amazon à votre client et à votre serveur de jeu avant de les déployer sur AmazonGameLift. Pour consulter la feuille de route complète qui vous permettra de lancer votre jeu sur AmazonGameLift, consultezFeuille de route d'hébergement GameLift géré par Amazon.

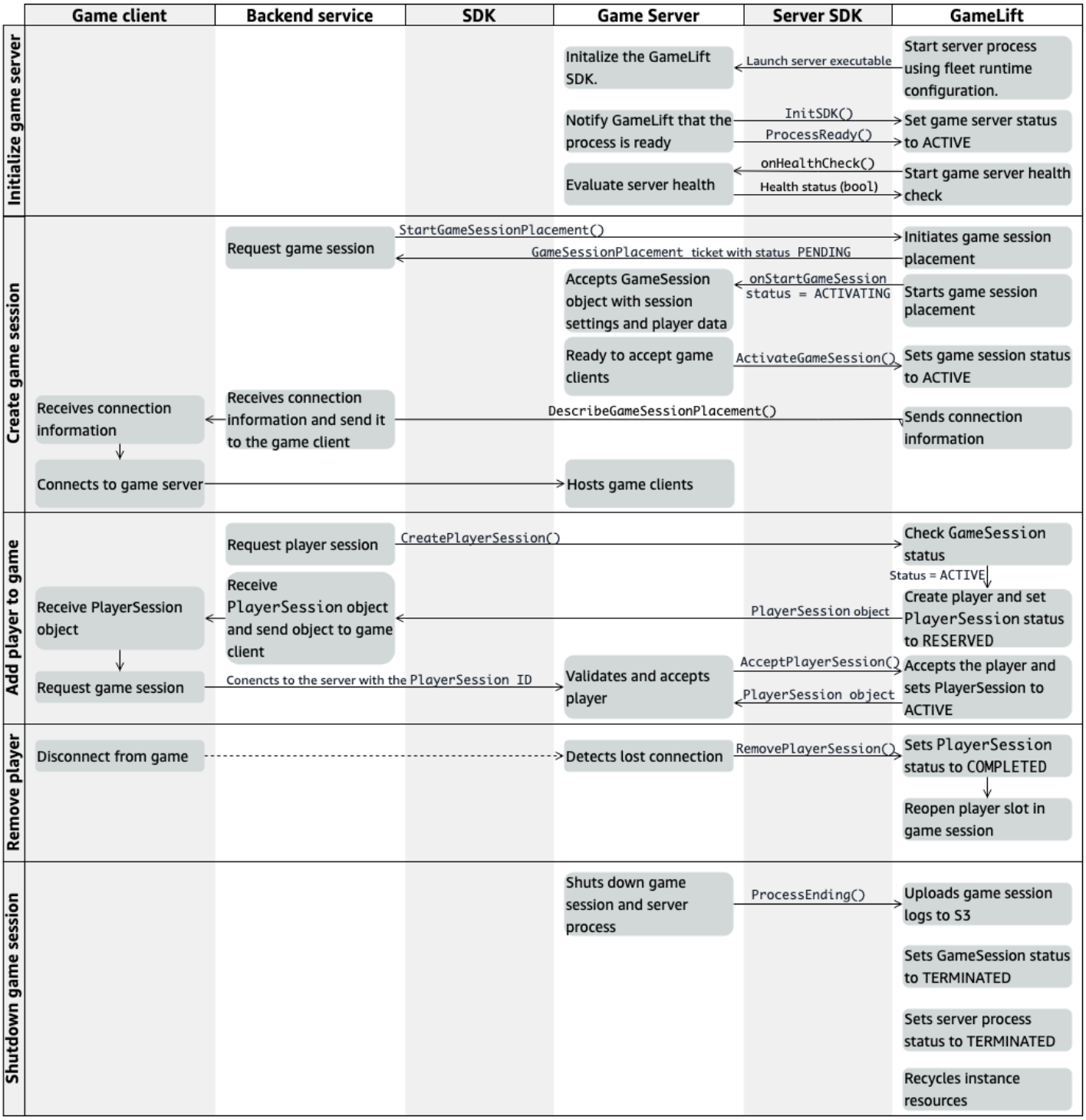
Rubriques

- Testez votre intégration à l'aide d'Amazon GameLift Local

Interactions entre Amazon GameLift et le serveur du jeu

Cette rubrique décrit les interactions entre le client du jeu, un service principal, un serveur de jeu et AmazonGameLift.

Le schéma suivant illustre les interactions entre le client du jeu, le service principal, le GameLift SDK Amazon, le serveur de jeu EC2 géré, le SDK du GameLift serveur Amazon et Amazon. GameLift Pour une description détaillée des interactions présentées, consultez les sections suivantes de cette page.



Initialiser un serveur de jeu

Les étapes suivantes décrivent les interactions qui se produisent lorsque vous préparez votre serveur de jeu à héberger des sessions de jeu.

1. Amazon GameLift lance l'exécutable du serveur sur une instance Amazon Elastic Compute Cloud (Amazon EC2).
2. Le serveur de jeu appelle :

   a. `InitSDK()` pour initialiser le kit SDK Server.

   b. `ProcessReady()`pour communiquer l'état de préparation de la session de jeu, les informations de connexion et l'emplacement des fichiers journaux des sessions de jeu.

   Le processus du serveur attend ensuite un rappel d'Amazon. GameLift
3. Amazon GameLift met à jour l'état du processus du serveur pour `ACTIVE` permettre le placement des sessions de jeu.
4. Amazon GameLift commence à appeler le `onHealthCheck` rappel et continue de l'appeler régulièrement pendant que le processus du serveur est actif. Le processus du serveur peut signaler qu'il est sain ou non en moins d'une minute.

Créer une session de jeu

Une fois que vous avez initialisé votre serveur de jeu, les interactions suivantes se produisent lorsque vous créez des sessions de jeu pour héberger vos joueurs.

1. Le service principal appelle l'opération du SDK. `StartGameSessionPlacement()`
2. Amazon GameLift crée un nouveau `GameSessionPlacement` ticket avec le statut `PENDING` et le renvoie au service principal.
3. Le service principal obtient le statut d'un ticket de placement à partir d'une file d'attente. Pour plus d'informations, veuillez consulter Configurer une notification d'événement pour le placement des sessions de jeu.
4. Amazon GameLift commence par placer les sessions de jeu en sélectionnant une flotte appropriée et en recherchant un processus de serveur actif dans une flotte contenant des sessions de `0` jeu. Lorsqu'Amazon GameLift localise un processus serveur, Amazon GameLift effectue les opérations suivantes :

   a. Crée un `GameSession` objet avec les paramètres de session de jeu et les données du joueur à partir de la demande de placement avec un `ACTIVATING` statut.

   b. Invoque le onStartGameSession rappel sur le processus serveur. Amazon GameLift transmet des informations à l'GameSessionobjet indiquant que le processus du serveur peut configurer la session de jeu.

   c. Modifie le nombre de sessions de jeu du processus serveur en1.

5. Le processus serveur exécute la fonction de onStartGameSession rappel. Lorsque le processus serveur est prêt à accepter les connexions des joueurs, il appelle ActivateGameSession() et attend les connexions des joueurs.

6. Amazon GameLift met à jour l'GameSessionobjet avec les informations de connexion pour le processus serveur. (Ces informations incluent le paramètre de port qui a été signalé avecProcessReady().) Amazon change GameLift également le statut enACTIVE.

7. Le service principal appelle DescribeGameSessionPlacement() pour détecter l'état mis à jour du ticket. Le service principal utilise ensuite les informations de connexion pour connecter le client du jeu au processus serveur et rejoindre la session de jeu.

Ajouter un joueur à un jeu

Cette séquence décrit le processus d'ajout d'un joueur à une session de jeu existante. Les sessions des joueurs peuvent également être demandées dans le cadre d'une demande de placement de session de jeu.

1. Le service principal appelle l'opération de l'API client à l'CreatePlayerSession()aide d'un identifiant de session de jeu.

2. Amazon GameLift vérifie l'état de la session de jeu (doit l'êtreACTIVE) et recherche un emplacement de joueur ouvert pendant la session de jeu. Si un emplacement est disponible, Amazon GameLift procède comme suit :

   a. Crée un nouvel PlayerSession objet et définit son statut surRESERVED.

   b. Répond à la demande de service principal avec l'PlayerSessionobjet.

3. Le service principal connecte le client du jeu directement au processus du serveur à l'aide de l'identifiant de session du joueur.

4. Le serveur appelle l'opération de l'API du serveur AcceptPlayerSession() pour valider l'ID de session du joueur. En cas de validation, Amazon GameLift transmet l'PlayerSessionobjet au processus du serveur. Le processus serveur accepte ou refuse la connexion.

5. Amazon GameLift effectue l'une des opérations suivantes :

a. Si la connexion est acceptée, Amazon GameLift définit le `PlayerSession` statut sur`ACTIVE`.

b. Si aucune réponse n'est reçue dans les 60 secondes suivant l'`CreatePlayerSession()`appel initial du serveur principal, Amazon GameLift change le `PlayerSession` statut `TIMEDOUT` et ouvre à nouveau l'emplacement du joueur pendant la session de jeu.

Supprimer un joueur

Lorsque vous supprimez des joueurs d'une session de jeu pour créer de l'espace pour que de nouveaux joueurs puissent y participer, les interactions suivantes se produisent.

1. Un joueur se déconnecte du jeu.
2. Le serveur détecte la perte de connexion et appelle l'opération de l'API du serveur`RemovePlayerSession()`.
3. Amazon GameLift change le `PlayerSession` statut `COMPLETED` et ouvre à nouveau l'emplacement du joueur pendant la session de jeu.

Arrêter la session de jeu

Cette séquence d'interactions se produit lorsqu'un processus serveur arrête la session de jeu en cours.

1. Le serveur ferme la session de jeu et le serveur.
2. Le serveur appelle `ProcessEnding()` AmazonGameLift.
3. Amazon GameLift effectue les opérations suivantes :

   a. Importe les journaux des sessions de jeu vers Amazon Simple Storage Service (Amazon S3).

   b. Remplace le `GameSession` statut par`TERMINATED`.

   c. Modifie l'état du processus du serveur en`TERMINATED`.

   d. Recycle les ressources de l'instance.

Intégrez votre serveur de jeu à Amazon GameLift

Une fois que votre serveur de jeu personnalisé est déployé et exécuté sur GameLift les instances Amazon, il doit être en mesure d'interagir avec Amazon GameLift (et potentiellement avec d'autres ressources). Cette section explique comment intégrer le logiciel de votre serveur de jeu à AmazonGameLift.

> **Note**
>
> Ces instructions supposent que vous avez créé un projet de serveur de jeu Compte AWS et que vous possédez déjà un projet de serveur de jeu.

Les rubriques de cette section décrivent comment gérer les tâches d'intégration suivantes :

- Établissez la communication entre Amazon GameLift et vos serveurs de jeux.
- Générez et utilisez un certificat TLS pour établir une connexion sécurisée entre le client et le serveur de jeu.
- Autorisez le logiciel de votre serveur de jeu à interagir avec d'autres AWS ressources.
- Autorisez les processus des serveurs de jeu à obtenir des informations sur la flotte sur laquelle ils fonctionnent.

Rubriques

Ajoutez Amazon GameLift à votre serveur de jeu

Votre serveur de jeu personnalisé doit communiquer avec AmazonGameLift, car chaque processus du serveur de jeu doit être en mesure de répondre aux événements déclenchés GameLift par Amazon. Votre serveur de jeu doit également GameLift informer Amazon de l'état des processus du serveur et des connexions des joueurs. Pour plus d'informations sur la façon dont votre serveur de jeu, votre service principal, votre client de jeu et Amazon GameLift collaborent pour gérer l'hébergement de jeux, consultezInteractions entre Amazon GameLift et le serveur du jeu.

Pour préparer votre serveur de jeu à interagir avec AmazonGameLift, ajoutez le SDK Amazon GameLift Server à votre projet de serveur de jeu et intégrez les fonctionnalités décrites dans cette

rubrique. Le SDK pour serveurs est disponible en plusieurs langues. Pour plus d'informations sur le SDK Amazon GameLift Server, consultezAssistance au développement avec Amazon GameLift.

Références de l'API du SDK pour serveurs :

- Référence GameLift du SDK Amazon Server 5.x pour C++
- Référence GameLift du SDK Amazon Server 5.x pour C# et Unity
- Référence du SDK GameLift 5.x du serveur Amazon Unreal Engine

Initialiser le processus du serveur

Ajoutez du code pour établir la communication avec Amazon GameLift et pour signaler que le processus du serveur est prêt à héberger une session de jeu. Ce code doit être exécuté avant tout GameLift code Amazon.

1. Initialisez le client d'GameLiftAPI Amazon en appelant`InitSdk()`. Pour initialiser un processus serveur sur une ressource de GameLift Anywhere calcul Amazon, appelez `InitSdk()` avec le numéro suivant : `ServerParameters`

   - L'URL du websocket utilisé pour se connecter à votre serveur de jeu.
   - L'ID du processus utilisé pour héberger votre serveur de jeu.
   - L'ID de l'ordinateur hébergeant les processus de votre serveur de jeu.
   - L'ID de la GameLift flotte contenant votre GameLift Anywhere ordinateur Amazon.
   - Le jeton d'autorisation généré par l'GameLiftopération Amazon`GetComputeAuthToken`.

   > **Note**
   >
   > Pour initialiser un serveur de jeu sur une instance Amazon EC2 GameLift gérée par Amazon, créez-le à ServerParameters l'aide du constructeur par défaut `InitSDK()` (C++) (C#) (Unreal) (C++) . Amazon GameLift configure l'environnement informatique et se connecte automatiquement à Amazon GameLift pour vous.

2. Avertissez Amazon GameLift qu'un processus serveur est prêt à héberger une session de jeu. Appelez `ProcessReady()` (C++) (C#) (Unreal) (C++) avec les informations suivantes. (Notez que vous ne devez appeler `ProcessReady()` qu'une seule fois par processus serveur).

- Le numéro de port utilisé par le processus serveur. Le service principal fournit le numéro de port et une adresse IP aux clients du jeu pour qu'ils se connectent au processus du serveur et rejoignent une session de jeu.
- L'emplacement des fichiers, tels que les journaux des sessions de jeu, que vous souhaitez qu'Amazon GameLift conserve. Le processus du serveur génère ces fichiers au cours d'une session de jeu. Ils sont temporairement stockés sur l'instance où s'exécute le processus du serveur et ils sont perdus lorsque l'instance s'arrête. Tous les fichiers que vous mettez en vente sont chargés sur AmazonGameLift. Vous pouvez accéder à ces fichiers via la GameLiftconsole Amazon ou en appelant l'opération d'GameLiftAPI Amazon GetGameSessionLogUrl().
- Les noms des fonctions de rappel qu'Amazon GameLift peut appeler vers votre processus serveur. Votre serveur de jeu doit implémenter ces fonctions. Pour plus d'informations, consultez (C++) (C#) (Unreal) (C++).
  - (Facultatif) `onHealthCheck` — Amazon GameLift appelle régulièrement cette fonction pour demander un rapport d'état de santé au serveur.
  - `onStartGameSession`— Amazon GameLift appelle cette fonction en réponse à la demande du client CreateGameSession().
  - `onProcessTerminate`— Amazon GameLift force le processus du serveur à s'arrêter, le laissant s'arrêter gracieusement.
  - (Facultatif) `onUpdateGameSession` — Amazon GameLift fournit un objet de session de jeu mis à jour au serveur de jeu ou fournit une mise à jour du statut d'une demande de remplissage de matchs. La fonction de FlexMatchremblayage nécessite ce rappel.

Vous pouvez également configurer un serveur de jeu pour accéder en toute sécurité aux AWS ressources que vous possédez ou contrôlez. Pour plus d'informations, veuillez consulter Communiquez avec les autres AWS ressources de vos flottes.

(Facultatif) État du processus du serveur de rapports

Ajoutez du code à votre serveur de jeu pour implémenter la fonction de rappel. `onHealthCheck()` Amazon GameLift invoque régulièrement cette méthode de rappel pour collecter des indicateurs de santé. Pour implémenter cette fonction de rappel, procédez comme suit :

- Évaluez l'état de santé du processus du serveur. Par exemple, vous pouvez signaler que le processus du serveur est défectueux en cas de défaillance d'une dépendance externe.

- Terminez l'évaluation du statut et répondez au rappel dans les 60 secondes. Si Amazon GameLift ne reçoit pas de réponse dans ce délai, il considère automatiquement que le processus du serveur est défectueux.
- Renvoie une valeur booléenne : true pour sain, false pour malsain.

Si vous n'implémentez pas de rappel pour vérifier l'état de santé, Amazon GameLift considère que le processus du serveur est sain à moins que le serveur ne réponde pas.

Amazon GameLift utilise l'intégrité des processus du serveur pour mettre fin aux processus défectueux et libérer des ressources. Si un processus serveur continue d'être signalé comme étant défectueux ou ne répond pas pendant trois contrôles d'intégrité consécutifs, Amazon GameLift peut arrêter le processus et en démarrer un nouveau. Amazon GameLift collecte des statistiques sur l'état des processus de serveurs d'un parc.

(Facultatif) Obtenir un certificat TLS

Si le processus du serveur s'exécute sur un parc sur lequel la génération de certificats TLS est activée, vous pouvez récupérer le certificat TLS pour établir une connexion sécurisée avec un client de jeu et pour crypter les communications client-serveur. Une copie du certificat est stockée sur l'instance. Pour obtenir l'emplacement du fichier, appelez `GetComputeCertificate( )`(C++) (C#) (Unreal) (C++).

Démarrez une session de jeu

Ajoutez du code permettant d'implémenter la fonction de rappel `onStartGameSession`. Amazon GameLift invoque ce rappel pour démarrer une session de jeu sur le serveur.

La `onStartGameSession` fonction prend un GameSessionobjet comme paramètre d'entrée. Cet objet inclut des informations clés sur la session de jeu, telles que le nombre maximum de joueurs. Il peut également inclure des données de jeu et des données de joueurs. L'implémentation de la fonction doit effectuer les tâches suivantes :

- Lancez des actions pour créer une nouvelle session de jeu en fonction des `GameSession` propriétés. Le serveur de jeu doit au minimum associer l'identifiant de session de jeu auquel les clients de jeu font référence lorsqu'ils se connectent au processus du serveur.
- Traitez les données de jeu et les données des joueurs selon vos besoins. Ces données se trouvent dans l'`GameSession`objet.

- Avertissez Amazon GameLift lorsqu'une nouvelle session de jeu est prête à accepter des joueurs. Appelez l'opération d'API du serveur `ActivateGameSession()`(C++) (C#) (Unreal) (C++). En réponse à un appel réussi, Amazon GameLift modifie le statut de la session de jeu en`ACTIVE`.

(Facultatif) Validez un nouveau joueur

Si vous suivez l'état des sessions des joueurs, ajoutez un code pour valider un nouveau joueur lorsqu'il se connecte à un serveur de jeu. Amazon GameLift suit les joueurs actuels et les créneaux de session de jeu disponibles.

Pour la validation, un client de jeu demandant l'accès à la session de jeu doit inclure un identifiant de session de joueur. Amazon génère GameLift automatiquement cet identifiant lorsqu'un joueur demande à rejoindre une partie en utilisant StartGameSessionPlacement() ou StartMatchmaking(). La session du joueur réserve ensuite un emplacement libre dans une session de jeu.

Lorsque le processus du serveur de jeu reçoit une demande de connexion client de jeu, il appelle `AcceptPlayerSession()` (C++) (C#) (Unreal) (C++) avec l'identifiant de session du joueur. En réponse, Amazon GameLift vérifie que l'identifiant de session du joueur correspond à un emplacement libre réservé lors de la session de jeu. Une fois qu'Amazon a GameLift validé l'identifiant de session du joueur, le processus du serveur accepte la connexion. Le joueur peut ensuite rejoindre la session de jeu. Si Amazon GameLift ne valide pas l'identifiant de session du joueur, le processus du serveur refuse la connexion.

(Facultatif) Signaler la fin d'une session de joueur

Si vous suivez l'état des sessions des joueurs, ajoutez un code pour avertir Amazon GameLift lorsqu'un joueur quitte la session de jeu. Ce code doit s'exécuter chaque fois que le processus serveur détecte une connexion abandonnée. Amazon GameLift utilise cette notification pour suivre les joueurs actuels et les machines à sous disponibles pendant la session de jeu.

Pour gérer les connexions interrompues, ajoutez dans votre code un appel à l'opération d'API du serveur `RemovePlayerSession()` (C++) (C#) (Unreal) (C++) avec l'identifiant de session du joueur correspondant.

Mettre fin à une session de jeu

Ajoutez du code à la séquence d'arrêt du processus du serveur pour GameLift avertir Amazon de la fin d'une session de jeu. Pour recycler et actualiser les ressources d'hébergement, Amazon GameLift arrête les processus du serveur une fois la session de jeu terminée.

Au début du code d'arrêt du processus serveur, appelez l'opération d'API du serveur `ProcessEnding()`(C++) (C#) (Unreal) (C++). Cet appel indique à Amazon GameLift que le processus du serveur est en train de s'arrêter. Amazon GameLift modifie l'état de la session de jeu et l'état du processus du serveur en`TERMINATED`. Après l'appel`ProcessEnding()`, le processus peut s'arrêter en toute sécurité.

Répondre à une notification d'arrêt d'un processus serveur

Ajoutez du code pour arrêter le processus du serveur en réponse à une notification d'AmazonGameLift. Amazon GameLift envoie cette notification lorsque le processus serveur signale régulièrement des dysfonctionnements ou si l'instance sur laquelle le processus serveur est en cours d'exécution est en cours d'arrêt. Amazon GameLift peut arrêter une instance dans le cadre d'un événement de réduction de capacité ou en réponse à l'interruption d'une instance Spot.

Pour gérer une notification d'arrêt, apportez les modifications suivantes au code de votre serveur de jeu :

- Implémentez la fonction de rappel `onProcessTerminate()`. Cette fonction doit appeler le code qui arrête le processus serveur. Lorsqu'Amazon GameLift invoque cette opération, les interruptions de l'instance Spot fournissent un préavis de deux minutes. Cet avis donne au serveur le temps nécessaire pour déconnecter correctement les joueurs, préserver les données relatives à l'état du jeu et effectuer d'autres tâches de nettoyage.
- Appelez l'opération d'API du serveur `GetTerminationTime()` (C++) (C#) (Unreal) (C++) de votre serveur de jeu. Si Amazon GameLift a lancé un appel pour arrêter le processus du serveur, il `GetTerminationTime()` renvoie l'heure de fin estimée.
- Au début du code d'arrêt de votre serveur de jeu, appelez l'opération d'API du serveur `ProcessEnding()`(C++) (C#) (Unreal) (C++). Cet appel indique à Amazon GameLift que le processus du serveur est en train de s'arrêter, et Amazon change GameLift alors l'état du processus du serveur en. `TERMINATED` Après l'appel`ProcessEnding()`, le processus peut s'arrêter en toute sécurité.

Communiquez avec les autres AWS ressources de vos flottes

Lorsque vous créez une version de serveur de jeu à déployer sur des GameLift flottes Amazon, vous souhaiterez peut-être que les applications de votre version de jeu communiquent directement et en toute sécurité avec les autres AWS ressources que vous possédez. Amazon GameLift gérant vos flottes d'hébergement de jeux, vous devez lui accorder un accès GameLift limité à ces ressources et services.

Voici quelques exemples de scénarios :

- Utilisez un CloudWatch agent Amazon pour collecter des métriques, des journaux et des traces à partir de flottes et de flottes EC2 gérées Anywhere
- Envoyez les données du journal de l'instance à Amazon CloudWatch Logs.
- Obtenez des fichiers de jeu stockés dans un bucket Amazon Simple Storage Service (Amazon S3).
- Lisez et écrivez des données de jeu (telles que les modes de jeu ou l'inventaire) stockées dans une base de données Amazon DynamoDB ou dans un autre service de stockage de données.
- Envoyez des signaux directement à une instance à l'aide d'Amazon Simple Queue Service (Amazon SQS).
- Accédez à des ressources personnalisées déployées et exécutées sur Amazon Elastic Compute Cloud (Amazon EC2).

Amazon GameLift prend en charge les méthodes suivantes pour établir l'accès :

- Accédez aux AWS ressources avec un rôle IAM
- Accédez aux AWS ressources grâce au peering VPC

Accédez aux AWS ressources avec un rôle IAM

Utilisez un rôle IAM pour spécifier qui peut accéder à vos ressources et définir des limites à cet accès. Les parties de confiance peuvent « assumer » un rôle et obtenir des informations de sécurité temporaires les autorisant à interagir avec les ressources. Lorsque les parties font des demandes d'API liées à la ressource, elles doivent inclure les informations d'identification.

Pour configurer le contrôle d'accès par un rôle IAM, effectuez les tâches suivantes :

1. Création du rôle IAM
2. Modifier les applications pour obtenir des informations d'identification
3. Associer une flotte au rôle IAM

Création du rôle IAM

Au cours de cette étape, vous créez un rôle IAM, avec un ensemble d'autorisations pour contrôler l'accès à vos AWS ressources et une politique de confiance qui donne à Amazon le GameLift droit d'utiliser les autorisations du rôle.

Pour obtenir des instructions sur la façon de configurer le rôle IAM, consultezConfigurer un rôle de service IAM pour Amazon GameLift. Lors de la création de la politique d'autorisations, choisissez les services, les ressources et les actions spécifiques que vos applications doivent utiliser. Il est recommandé de limiter autant que possible l'étendue des autorisations.

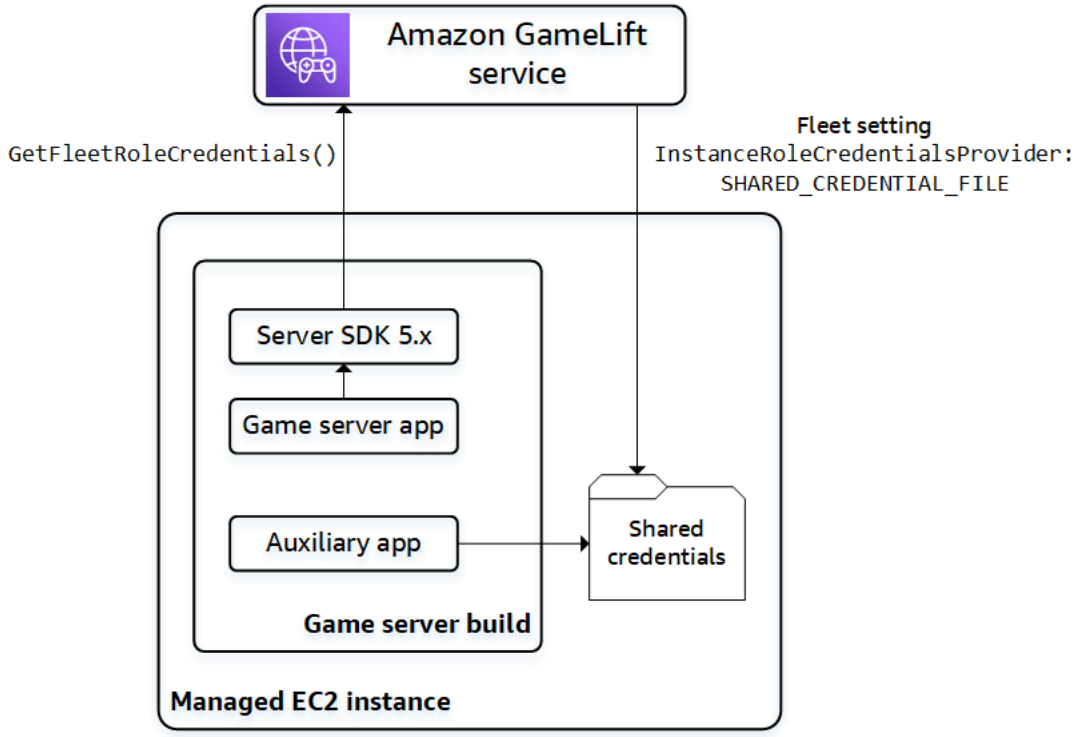
Après avoir créé le rôle, prenez note du nom de ressource Amazon (ARN) du rôle. Vous avez besoin du rôle ARN lors de la création de la flotte.

Modifier les applications pour obtenir des informations d'identification

Au cours de cette étape, vous configurez vos applications pour obtenir des informations d'identification de sécurité pour le rôle IAM et les utiliser lors de l'interaction avec vos AWS ressources. Consultez le tableau suivant pour déterminer comment modifier vos applications en fonction (1) du type d'application et (2) de la version du SDK du serveur que votre jeu utilise pour communiquer avec Amazon GameLift.

	Applications pour serveurs de jeux	Autres applications
Utilisation du SDK du serveur version 5.x	Appelez la méthode SDK du serveur `GetFleetRoleCredentials()` à partir du code de votre serveur de jeu.	Ajoutez du code à l'application pour extraire les informations d'identif ication d'un fichier partagé sur l'instance de flotte.
Utilisation du SDK du serveur version 4 ou antérieure	Appelez AWS Security Token Service (AWS STS) `AssumeRole` avec le rôle ARN.	Appelez AWS Security Token Service (AWS STS) `AssumeRole` avec le rôle ARN.

Pour les jeux intégrés au SDK 5.x du serveur, ce schéma montre comment les applications de votre version de jeu déployée peuvent acquérir des informations d'identification pour le rôle IAM.



Appel **`GetFleetRoleCredentials()`** (SDK 5.x du serveur)

Dans le code de votre serveur de jeu, qui devrait déjà être intégré au SDK 5.x GameLift du serveur Amazon, appelez `GetFleetRoleCredentials` (C++) (C#) (Unreal) pour récupérer un ensemble d'informations d'identification temporaires. Lorsque les informations d'identification expirent, vous pouvez les actualiser avec un autre appel à`GetFleetRoleCredentials`.

Utiliser des informations d'identification partagées (SDK du serveur 5.x)

Pour les applications non serveur déployées avec des versions de serveurs de jeu à l'aide du SDK 5.x du serveur, ajoutez du code pour obtenir et utiliser les informations d'identification stockées dans un fichier partagé. Amazon GameLift génère un profil d'informations d'identification pour chaque instance de flotte. Les informations d'identification peuvent être utilisées par toutes les applications de l'instance. Amazon actualise GameLift en permanence les informations d'identification temporaires.

Vous devez configurer une flotte pour générer le fichier d'informations d'identification partagé lors de la création de la flotte.

Dans chaque application qui doit utiliser le fichier d'informations d'identification partagé, spécifiez l'emplacement du fichier et le nom du profil, comme suit :

Windows :

```
[credentials]
shared_credential_profile= "FleetRoleCredentials"
shared_credential_file= "C:\\Credentials\\credentials"
```

Linux :

```
[credentials]
shared_credential_profile= "FleetRoleCredentials"
shared_credential_file= "/local/credentials/credentials"
```

Exemple : configurer un CloudWatch agent pour collecter des métriques pour les instances de GameLift flotte Amazon

Si vous souhaitez utiliser un CloudWatch agent Amazon pour collecter des métriques, des journaux et des traces à partir de vos GameLift flottes Amazon, utilisez cette méthode pour autoriser l'agent à émettre les données vers votre compte. Dans ce scénario, suivez les étapes suivantes :

1. Récupérez ou écrivez le `config.json` fichier de CloudWatch l'agent.
2. Mettez à jour le `common-config.toml` fichier pour que l'agent identifie le nom du fichier d'informations d'identification et le nom du profil, comme décrit ci-dessus.
3. Configurez le script d'installation de build de votre serveur de jeu pour installer et démarrer l' CloudWatch agent.

Utilisation **`AssumeRole()`** (SDK 4 du serveur)

Ajoutez du code à vos applications pour assumer le rôle IAM et obtenir des informations d'identification pour interagir avec vos AWS ressources. Toute application qui s'exécute sur une instance Amazon GameLift Fleet avec le SDK de serveur 4 ou une version antérieure peut assumer le rôle IAM.

Dans le code de l'application, avant d'accéder à une AWS ressource, l'application doit appeler l'opération `AssumeRole` API AWS Security Token Service (AWS STS) et spécifier l'ARN du rôle. Cette opération renvoie un ensemble d'informations d'identification temporaires qui autorisent l'application à accéder à la AWS ressource. Pour plus d'informations, consultez Utilisation d'informations d'identification temporaires avec des ressources AWS dans le Guide de l'utilisateur IAM.

Associer une flotte au rôle IAM

Après avoir créé le rôle IAM et mis à jour les applications de la version de votre serveur de jeu pour obtenir et utiliser les informations d'accès, vous pouvez déployer une flotte. Lorsque vous configurez la nouvelle flotte, définissez les paramètres suivants :

- InstanceRoleArn— Définissez ce paramètre sur l'ARN du rôle IAM.
- InstanceRoleCredentialsProvider— Pour demander GameLift à Amazon de générer un fichier d'informations d'identification partagé pour chaque instance de flotte, définissez ce paramètre sur`SHARED_CREDENTIAL_FILE`.

Vous devez définir ces valeurs lors de la création de la flotte. Ils ne pourront pas être mis à jour ultérieurement.

Accédez aux AWS ressources grâce au peering VPC

Vous pouvez utiliser le peering Amazon Virtual Private Cloud (Amazon VPC) pour communiquer entre les applications exécutées sur une GameLift instance Amazon et une autre ressource. AWS Un VPC est un réseau privé virtuel que vous définissez et qui inclut un ensemble de ressources gérées via votre. Compte AWS Chaque GameLift flotte Amazon possède son propre VPC. Avec le peering VPC, vous pouvez établir une connexion réseau directe entre le VPC de votre flotte et celui de vos autres ressources. AWS

Amazon GameLift rationalise le processus de configuration des connexions de peering VPC pour vos serveurs de jeux. Il gère les demandes d'appairage, met à jour les tables de routage et configure les connexions requises. Pour obtenir des instructions sur la façon de configurer le peering VPC pour vos serveurs de jeu, consultez. Peering VPC pour Amazon GameLift

Intégrez votre client de jeu à Amazon GameLift

Les rubriques de cette section décrivent les GameLift fonctionnalités Amazon gérées que vous pouvez ajouter à un service principal. Un service principal gère les tâches suivantes :

- Demande des informations sur les sessions de jeu actives auprès d'AmazonGameLift.
- Connecte un joueur à une session de jeu existante.
- Crée une nouvelle session de jeu et associe les joueurs à celle-ci.
- Modifie les métadonnées d'une session de jeu existante.

Pour plus d'informations sur la manière dont les clients de jeux interagissent avec Amazon GameLift et les serveurs de jeux exécutés sur AmazonGameLift, consultezInteractions entre Amazon GameLift et le serveur du jeu.

Prérequis

- Un Compte AWS.
- Une version de serveur de jeu téléchargée sur AmazonGameLift.
- Une flotte pour héberger vos jeux.

Rubriques

- Ajoutez Amazon GameLift à votre client de jeu
- Générer des identifiants de joueurs

Ajoutez Amazon GameLift à votre client de jeu

Intégrez Amazon GameLift aux composants de jeu qui nécessitent des informations sur les sessions de jeu, créez de nouvelles sessions de jeu et ajoutez des joueurs aux jeux. Selon l'architecture de votre jeu, cette fonctionnalité se trouve dans les services principaux qui gèrent des tâches telles que l'authentification des joueurs, le matchmaking ou le placement des sessions de jeu.

> **Note**
>
> Pour obtenir des informations détaillées sur la façon de configurer le matchmaking pour votre jeu GameLift hébergé par Amazon, consultez le guide du GameLift FlexMatch développeur Amazon.

Configurer Amazon GameLift sur un service de backend

Ajoutez du code pour initialiser un GameLift client Amazon et enregistrer les paramètres clés. Ce code doit être exécuté avant tout code dépendant d'Amazon GameLift.

1. Configurez une configuration client. Utilisez la configuration client par défaut ou créez un objet de configuration client personnalisé. Pour plus d'informations, consultez AWS::Client::ClientConfiguration(C++) ou AmazonGameLiftConfig(C#).

Une configuration client spécifie une région cible et un point de terminaison à utiliser pour contacter Amazon GameLift. La région identifie l'ensemble des ressources déployées (flottes, files d'attente et entremetteurs) à utiliser. La configuration du client par défaut définit l'emplacement sur la région USA Est (Virginie du Nord). Pour utiliser une autre région, créez une configuration personnalisée.

2. Initialisez un GameLift client Amazon. Utilisez Aws : GameLift : : GameLiftClient () (C++) ou AmazonGameLiftClient() (C#) avec une configuration client par défaut ou une configuration client personnalisée.
3. Ajoutez un mécanisme permettant de générer un identifiant unique pour chaque joueur. Pour plus d'informations, consultez Générer des identifiants de joueurs.
4. Collectez et stockez les informations suivantes :

   - Parc cible : de nombreuses demandes GameLift d'API Amazon doivent spécifier un parc. Pour ce faire, utilisez un identifiant de flotte ou un identifiant d'alias pointant vers le parc cible. Il est recommandé d'utiliser des alias de flotte afin de pouvoir changer de joueur d'une flotte à l'autre sans avoir à mettre à jour vos services principaux.
   - File d'attente cible : pour les jeux qui utilisent des files d'attente impliquant plusieurs flottes pour créer de nouvelles sessions de jeu, spécifiez le nom de la file d'attente à utiliser.
   - AWS informations d'identification : tous les appels vers Amazon GameLift doivent fournir les informations d'identification de Compte AWS l'hébergeur du jeu. Vous obtenez ces informations d'identification en créant un utilisateur joueur, comme décrit dansConfigurez l'accès programmatique à votre jeu. En fonction de la façon dont vous gérez l'accès de l'utilisateur du joueur, procédez comme suit :
     - Si vous utilisez un rôle pour gérer les autorisations des utilisateurs des joueurs, ajoutez du code pour assumer ce rôle avant d'appeler une GameLift API Amazon. La demande d'attribution du rôle renvoie un ensemble d'informations d'identification de sécurité temporaires. Pour plus d'informations, consultez la section Passage à un rôle IAM (AWS API) dans le guide de l'utilisateur IAM.
     - Si vous disposez d'informations d'identification de sécurité à long terme, configurez votre code pour localiser et utiliser les informations d'identification stockées. Voir Authentification à l'aide d'informations d'identification à long terme dans le AWS Guide de référence des SDK et des outils. Pour plus d'informations sur le stockage des informations d'identification, consultez les références d'AWS API pour (C++) et (.NET).
     - Si vous disposez d'informations d'identification de sécurité temporaires, ajoutez du code pour actualiser régulièrement les informations d'identification à l'aide du AWS Security

Token Service (AWS STS), comme décrit dans la section Utilisation d'informations d'identification de sécurité temporaires avec AWS les SDK du guide de l'utilisateur IAM. Le code doit demander de nouvelles informations d'identification avant que les anciennes n'expirent.

Obtenez des sessions de jeu

Ajoutez du code pour découvrir les sessions de jeu disponibles et gérer les paramètres et les métadonnées des sessions de jeu.

Rechercher des sessions de jeu actives

SearchGameSessionsUtilisez-le pour obtenir des informations sur une session de jeu spécifique, toutes les sessions actives ou les sessions répondant à un ensemble de critères de recherche. Cet appel renvoie un GameSessionobjet pour chaque session de jeu active correspondant à votre demande de recherche.

Utilisez des critères de recherche pour obtenir une liste filtrée des sessions de jeu actives auxquelles les joueurs peuvent se connecter. Par exemple, vous pouvez filtrer les sessions de la façon suivante :

- Exclure les sessions de jeu complètes :`CurrentPlayerSessionCount = MaximumPlayerSessionCount`.
- Choisissez les sessions de jeu en fonction de la durée pendant laquelle la session est en cours : Évaluez`CreationTime`.
- Trouvez des sessions de jeu en fonction d'une propriété de jeu personnalisée :`gameSessionProperties.gameMode = "brawl"`.

Gérer les sessions de jeu

Utilisez les opérations suivantes pour récupérer ou mettre à jour des informations sur des sessions de jeu.

- DescribeGameSessionDetails() — Obtenez le statut de protection d'une session de jeu en plus des informations de session de jeu.
- UpdateGameSession() — Modifiez les métadonnées et les paramètres d'une session de jeu selon vos besoins.
- GetGameSessionLogUrl— Accédez aux journaux de session de jeu stockés.

Créez des sessions de jeu

Ajoutez du code pour démarrer de nouvelles sessions de jeu sur vos flottes déployées et les mettre à disposition des joueurs. Il existe deux options pour créer des sessions de jeu, selon que vous déployez votre jeu dans plusieurs régions Régions AWS ou dans une seule.

Créez une session de jeu dans une file d'attente multisite

StartGameSessionPlacementÀ utiliser pour placer une demande de nouvelle session de jeu dans une file d'attente. Pour utiliser cette opération, créez une file d'attente. Cela détermine où Amazon GameLift place la nouvelle session de jeu. Pour plus d'informations sur les files d'attente et sur leur utilisation, consultezConfiguration des GameLift files d'attente Amazon pour le placement des sessions de jeu.

Lorsque vous créez un emplacement de session de jeu, spécifiez le nom de la file d'attente à utiliser, un nom de session de jeu, un nombre maximum de joueurs simultanés et un ensemble facultatif de propriétés de jeu. Vous pouvez également éventuellement fournir une liste de joueurs qui rejoindront automatiquement la session de jeu. Si vous incluez des données de latence des joueurs pour les régions concernées, Amazon GameLift utilise ces informations pour placer la nouvelle session de jeu dans une flotte offrant une expérience de jeu idéale aux joueurs.

Le placement de session de jeu est un processus asynchrone. Une fois que vous avez fait une demande, vous pouvez la laisser aboutir ou expirer. Vous pouvez également annuler la demande à tout moment en utilisant StopGameSessionPlacement. Pour vérifier le statut de votre demande de placement, appelez DescribeGameSessionPlacement.

Créez une session de jeu sur une flotte spécifique

CreateGameSessionÀ utiliser pour créer une nouvelle session sur un parc spécifique. Cette opération synchrone réussit ou échoue selon que la flotte dispose de ressources disponibles pour héberger une nouvelle session de jeu. Une fois qu'Amazon a GameLift créé la nouvelle session de jeu et renvoyé un GameSessionobjet, vous pouvez y associer des joueurs.

Lorsque vous utilisez cette opération, fournissez un identifiant de flotte ou un identifiant d'alias, un nom de session et le nombre maximum de joueurs simultanés pour le jeu. Vous pouvez également inclure un ensemble de propriétés de jeu. Les propriétés du jeu sont définies dans un tableau de paires clé-valeur.

Si vous utilisez la fonctionnalité de protection GameLift des ressources d'Amazon pour limiter le nombre de sessions de jeu qu'un joueur peut créer, fournissez l'identifiant de joueur du créateur de la session de jeu.

Joindre un joueur à une session de jeu

Ajoutez un code pour réserver une place de joueur dans une session de jeu active et connecter les clients du jeu aux sessions de jeu.

1. Réservez une place de joueur lors d'une session de jeu

   Pour réserver un emplacement de joueur, créez une session de joueur pour la session de jeu. Pour plus d'informations sur les sessions des joueurs, consultezComment les joueurs se connectent aux jeux.

   Il existe deux manières de créer de nouvelles sessions pour les joueurs :

   - StartGameSessionPlacementÀ utiliser pour réserver des places à un ou plusieurs joueurs lors de la nouvelle session de jeu.
   - Réservez des places pour un ou plusieurs joueurs en utilisant CreatePlayerSessionou CreatePlayerSessionsavec un identifiant de session de jeu.

   Amazon vérifie GameLift d'abord que la session de jeu accepte de nouveaux joueurs et qu'il y a des emplacements disponibles pour les joueurs. En cas de succès, Amazon GameLift réserve une place au joueur, crée la nouvelle session de joueur et renvoie un PlayerSessionobjet. Cet objet contient le nom DNS, l'adresse IP et le port dont un client de jeu a besoin pour se connecter à la session de jeu.

   Une demande de session de joueur doit inclure un ID unique pour chaque joueur. Pour plus d'informations, consultez Générer des identifiants de joueurs.

   Une session de joueur peut inclure un ensemble de données de joueur personnalisées. Ces données sont stockées dans le nouvel objet de session du joueur, que vous pouvez récupérer en appelant DescribePlayerSessions(). Amazon transmet GameLift également cet objet au serveur de jeu lorsque le joueur se connecte directement à la session de jeu. Lorsque vous demandez des sessions à plusieurs joueurs, fournissez une chaîne de données de joueur pour chaque joueur mappée à l'identifiant du joueur indiqué dans la demande.

2. Se connecter à une session de jeu

   Ajoutez du code au client de jeu pour récupérer l'objet `PlayerSession`, qui contient les informations de connexion de la session de jeu. Utilisez ces informations pour établir une connexion directe avec le serveur.

- Vous pouvez vous connecter à l'aide du port spécifié et du nom DNS ou de l'adresse IP attribués au processus serveur.
- Si la génération de certificats TLS est activée dans vos flottes, connectez-vous à l'aide du nom et du port DNS.
- Si votre serveur de jeu valide les connexions entrantes entre joueurs, faites référence à l'identifiant de session du joueur.

Une fois la connexion établie, le client du jeu et le processus serveur communiquent directement sans impliquer Amazon GameLift. Le serveur maintient la communication avec Amazon GameLift pour signaler l'état de connexion des joueurs, leur état de santé, etc. Si le serveur de jeu valide les joueurs entrants, il vérifie que l'identifiant de session du joueur correspond à un emplacement réservé dans la session de jeu et accepte ou refuse la connexion du joueur. Lorsque le joueur se déconnecte, le processus serveur signale l'interruption de la connexion.

Utiliser les propriétés des sessions de jeu

Votre client de jeu peut transmettre des données à une session de jeu en utilisant une propriété de jeu. Les propriétés du jeu sont des paires clé-valeur que votre serveur de jeu peut ajouter, lire, répertorier et modifier. Vous pouvez transmettre une propriété de jeu lorsque vous créez une nouvelle session de jeu, ou plus tard lorsque la session de jeu est active. Une session de jeu peut contenir jusqu'à 16 propriétés de jeu. Vous ne pouvez pas supprimer les propriétés du jeu.

Par exemple, votre jeu propose les niveaux de difficulté suivants : `NoviceEasy`,`Intermediate`, et`Expert`. Un joueur choisit`Easy`, puis commence la partie. Votre client de jeu demande une nouvelle session de jeu à Amazon GameLift en utilisant l'une des deux `CreateGameSession` options `StartGameSessionPlacement` ou comme expliqué dans les sections précédentes. Dans la demande, le client transmet ceci :`{"Key": "Difficulty", "Value":"Easy"}`.

En réponse à cette demande, Amazon GameLift crée un `GameSession` objet contenant la propriété de jeu spécifiée. Amazon demande GameLift ensuite à un serveur de jeu disponible de démarrer la nouvelle session de jeu et transmet l'`GameSession`objet. Le serveur de jeu démarre une session de jeu avec un `Difficulty` de`Easy`.

En savoir plus

- GameProperty type de données
- SearchGameSessions() exemples

- UpdateGameSession GameProperties paramètre ()

Générer des identifiants de joueurs

Amazon GameLift utilise une session de joueur pour représenter un joueur connecté à une session de jeu. Amazon GameLift crée une session joueur chaque fois qu'un joueur se connecte à une session de jeu à l'aide d'un client de jeu intégré à AmazonGameLift. Lorsqu'un joueur quitte une partie, la session du joueur prend fin. Amazon GameLift ne réutilise pas les sessions des joueurs.

L'exemple de code suivant génère des identifiants de joueur uniques de manière aléatoire :

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

Pour plus d'informations sur les sessions des joueurs, consultezAfficher les données sur les sessions de jeu et les sessions des joueurs.

Moteurs de jeu et Amazon GameLift

Vous pouvez utiliser le GameLift service Amazon géré avec la plupart des principaux moteurs de jeu qui prennent en charge les bibliothèques C++ ou C#, notamment O3DE, Unreal Engine et Unity. Générez la version dont vous avez besoin pour votre jeu. Pour en savoir plus sur les instructions de génération et les exigences minimales, consultez les fichiers README de chaque version. Pour plus d'informations sur les GameLift kits SDK Amazon disponibles, les plateformes de développement et les systèmes d'exploitation pris en charge, consultez la section consacrée Assistance au développement avec Amazon GameLift aux serveurs de jeux.

Outre les informations spécifiques au moteur fournies dans cette rubrique, vous trouverez de l'aide supplémentaire sur l'intégration d'Amazon GameLift à vos serveurs, clients et services de jeu dans les rubriques suivantes :

- Feuille de route d'hébergement GameLift géré par Amazon— Un flux de travail GameLift en six étapes pour intégrer avec succès Amazon à votre jeu et configurer des ressources d'hébergement.
- Ajoutez Amazon GameLift à votre serveur de jeu— Instructions détaillées sur l'intégration d'Amazon GameLift dans un serveur de jeu.

- Ajoutez Amazon GameLift à votre client de jeu : instructions détaillées sur l'intégration à un client de jeu ou un service, y compris la création de sessions de jeu et la participation de joueurs aux jeux.

O3DE

Serveurs de jeux

Préparez vos serveurs de jeu pour l'hébergement sur Amazon GameLift à l'aide GameLiftdu SDK Amazon Server pour C++. Consultez Ajoutez Amazon GameLift à votre serveur de jeu pour obtenir de l'aide sur l'intégration des fonctionnalités requises à votre serveur de jeux.

Clients et services de jeux

Permettez à vos clients et/ou services de jeu d'interagir avec le GameLift service Amazon, par exemple pour trouver des sessions de jeu disponibles ou en créer de nouvelles, et ajouter des joueurs à des jeux. Les fonctionnalités principales du client sont fournies dans le AWSSDK pour C++. Pour GameLift intégrer Amazon à votre projet de jeu O3DE, consultez Ajouter Amazon GameLift à un client et à un serveur de jeu O3DE et. Ajoutez Amazon GameLift à votre client de jeu

Unreal Engine

Serveurs de jeux

Préparez vos serveurs de jeu pour l'hébergement sur Amazon GameLift en ajoutant le SDK Amazon GameLift Server pour Unreal Engine à votre projet et en mettant en œuvre les fonctionnalités de serveur requises. Pour obtenir de l'aide sur la configuration du plugin Unreal Engine et l'ajout de GameLift code Amazon, consultezIntégrer Amazon GameLift dans un projet Unreal Engine.

Clients et services de jeux

Permettez à vos clients et/ou services de jeu d'interagir avec le GameLift service Amazon, par exemple pour trouver des sessions de jeu disponibles ou en créer de nouvelles, et ajouter des joueurs à des jeux. Les fonctionnalités principales du client sont fournies dans le AWSSDK pour C++. Pour GameLift intégrer Amazon à votre projet de jeu Unreal Engine, consultezAjoutez Amazon GameLift à votre client de jeu.

Unity

Serveurs de jeux

Préparez vos serveurs de jeu pour l'hébergement sur Amazon GameLift en ajoutant le SDK Amazon GameLift Server pour C# à votre projet et en mettant en œuvre les fonctionnalités de serveur requises. Pour obtenir de l'aide pour configurer Unity et ajouter GameLift du code Amazon, consultezIntégrer Amazon GameLift dans un projet Unity.

Clients et services de jeux

Permettez à vos clients et/ou services de jeu d'interagir avec le GameLift service Amazon, par exemple pour trouver des sessions de jeu disponibles ou en créer de nouvelles, et ajouter des joueurs à des jeux. Les principales fonctionnalités client sont fournies dans le kit AWS SDK for .NET. Pour GameLift intégrer Amazon à votre projet de jeu Unity, consultezAjoutez Amazon GameLift à votre client de jeu.

Autres moteurs

Pour obtenir la liste complète des GameLift kits de développement logiciel Amazon disponibles pour les serveurs et les clients de jeux, consultezthe section called “Soutien au développement”.

Ajouter Amazon GameLift à un client et à un serveur de jeu O3DE

Vous pouvez utiliser O3DE, un moteur 3D open source, multiplateforme et en temps réel pour créer des expériences interactives de haute performance, notamment des jeux et des simulations. Le moteur de rendu et les outils O3DE sont intégrés dans un cadre modulaire que vous pouvez modifier et étendre avec vos outils de développement préférés.

Le framework modulaire utilise Gems qui contiennent des bibliothèques avec des interfaces et des actifs standard. Sélectionnez vos propres gemmes pour choisir les fonctionnalités à ajouter en fonction de vos besoins.

L'Amazon GameLift Gem fournit les fonctionnalités suivantes :

GameLiftIntégration à Amazon

> Un framework pour étendre la couche réseau O3DE et permettre au Multiplayer Gem de fonctionner avec la solution de serveur GameLift dédié Amazon. The Gem fournit des intégrations à la fois avec le SDK GameLift du serveur Amazon et le client du AWS SDK (pour appeler le GameLift service Amazon lui-même).

Gestion des compilations et des packages

Instructions pour empaqueter et éventuellement télécharger la version du serveur dédié et une application AWS Cloud Development Kit (AWS CDK) (AWS CDK) pour configurer et mettre à jour les ressources.

Configuration d'Amazon GameLift Gem

Suivez les procédures décrites dans cette section pour configurer l'Amazon GameLift Gem dans O3DE.

Prérequis

- Configurez votre AWS compte pour AmazonGameLift. Pour plus d'informations, veuillez consulter Configurez un Compte AWS.
- Configurez les AWS informations d'identification pour O3DE. Pour plus d'informations, voir Configuration des AWS informations d'identification.
- Configurez le AWS CLI groupeAWS CDK. Pour plus d'informations, AWS Command Line Interfaceet AWS Cloud Development Kit (AWS CDK).

Activez Amazon GameLift Gem et ses dépendances

1. Ouvrez le gestionnaire de projets.
2. Ouvrez le menu situé sous votre projet et choisissez Modifier les paramètres du projet... .
3. Choisissez Configurer les gemmes.
4. Activez la GameLift gemme Amazon et les gemmes dépendantes suivantes :
   - AWSCore Gem : fournit le cadre à utiliser Services AWS dans O3DE.
   - Joyau multijoueur — Fournit des fonctionnalités multijoueurs en étendant l'infrastructure réseau.

Inclure la bibliothèque statique Amazon GameLift Gem

1. Incluez `Gem::AWSGameLift.Server.Static` les annonces `BUILD_DEPENDENCIES` correspondant à la cible de votre serveur de projet.

```
ly_add_target(
```

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

2. Définissez `AWSGameLiftService` ce paramètre sur requis pour le composant système de votre serveur de projet.

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

3. (Facultatif) Pour effectuer des demandes de GameLift service Amazon en C++, `Gem::AWSGameLift.Client.Static` incluez-le dans la cible `BUILD_DEPENDENCIES` pour votre client.

```
ly_add_target(
    NAME YourProject.Client.Static STATIC
    ...
    BUILD_DEPENDENCIES
    PUBLIC
        ...
    PRIVATE
        ...
        Gem::AWSCore.Static
        Gem::AWSGameLift.Client.Static
}
```

Intégrez votre jeu et votre serveur dédié

Gérez les sessions de jeu au sein de votre jeu et de votre serveur de jeu dédié grâce à l'intégration de gestion de session. Pour obtenir de FlexMatch l'aide, voir FlexMatchIntégration.

Intégrer Amazon GameLift dans un projet Unreal Engine

Cette rubrique explique comment configurer le plug-in SDK du serveur Amazon GameLift C++ pour Unreal Engine et l'intégrer dans vos projets de jeu.

Ressources supplémentaires :

- Plug-in SDK du serveur pour le site de téléchargement d'Unreal
- Référence du SDK GameLift 5.x du serveur Amazon Unreal Engine
- the section called “Soutien au développement”

Prérequis

Avant de continuer, assurez-vous d'avoir passé en revue les conditions préalables suivantes :

Prérequis

- Un ordinateur capable d'exécuter Unreal Engine. Pour plus d'informations sur les exigences d'Unreal Engine, consultez la documentation relative aux spécifications matérielles et logicielles d'Unreal Engine.
- Microsoft Visual Studio 2019 ou version plus récente.
- CMake version 3.1 ou ultérieure.
- Python version 3.6 ou ultérieure.
- Un client Git disponible sur le PATH.
- Un compte Epic Games. Ouvrez un compte sur le site officiel d'Unreal Engine.
- Un GitHub compte associé à votre compte Unreal Engine. Pour plus d'informations, consultez la section Accès au code source d'Unreal Engine GitHub sur le site Web d'Unreal Engine.

> **Note**
>
> Amazon prend GameLift actuellement en charge les versions suivantes d'Unreal Engine :
>
> - 4,22

- 4,23
- 4,24
- 4,25
- 4,26
- 4,27
- 5.1.0
- 5.1.1
- 5.2
- 5.3

Créez Unreal Engine à partir des sources

Les versions standard de l'éditeur Unreal Engine, téléchargées via le lanceur Epic, autorisent uniquement les builds d'applications clientes Unreal. Pour créer une application serveur Unreal, vous devez télécharger et créer Unreal Engine à partir des sources, en utilisant le référentiel Unreal Engine Github. Pour plus d'informations, consultez le didacticiel de création d'Unreal Engine à partir des sources sur le site Web de documentation d'Unreal Engine.

> **Note**
>
> Si ce n'est pas déjà fait, suivez les instructions de la section Accès au code source d'Unreal Engine GitHub pour associer votre GitHub compte à votre compte Epic Games.

Pour cloner le code source Unreal Engine dans votre environnement de développement

1. Clonez le code source Unreal Engine dans votre environnement de développement dans une branche de votre choix.

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

2. Consultez le tag de la version que vous utilisez pour développer votre jeu. Par exemple, l'exemple suivant utilise la version 5.1.1 d'Unreal Engine :

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

3. Accédez au dossier racine du référentiel local. Lorsque vous êtes dans le dossier racine, exécutez le fichier suivant :`Setup.bat`.
4. Dans le dossier racine, exécutez également le fichier :`GenerateProjectFiles.bat`.
5. Après avoir exécuté les fichiers des étapes précédentes, un fichier de solution Unreal Engine est créé. `UE5.sln` Ouvrez Visual Studio, puis ouvrez le `UE5.sln` fichier dans l'éditeur Visual Studio.
6. Dans Visual Studio, ouvrez le menu Affichage et choisissez l'option Explorateur de solutions. Cela ouvre le menu contextuel du nœud du projet Unreal. Dans la fenêtre de l'Explorateur de solutions, cliquez avec le bouton droit sur le `UE5.sln` fichier (il peut être simplement répertoriéUE5), puis choisissez Build pour créer le projet Unreal avec la cible Win64 de l'éditeur de développement.

   > **Note**
   >
   > La construction peut prendre plus d'une heure.

Une fois la compilation terminée, vous êtes prêt à ouvrir l'éditeur de développement Unreal et à créer ou importer un projet.

Configurez votre projet Unreal pour le plugin

Suivez ces étapes pour préparer le plug-in SDK Amazon GameLift Server pour Unreal Engine à vos projets de serveur de jeu.

Pour configurer votre projet pour le plugin

1. Visual Studio étant ouvert, accédez au volet Explorateur de solutions et choisissez le `UE5` fichier pour ouvrir le menu contextuel du projet Unreal. Dans le menu contextuel, choisissez l'option Définir comme projet de démarrage.
2. En haut de votre fenêtre Visual Studio, choisissez Démarrer le débogage (flèche verte).

   Cette action démarre votre nouvelle instance créée en source d'Unreal Editor. Pour plus d'informations sur l'utilisation de l'éditeur Unreal, consultez l'interface Unreal Editor sur le site Web de documentation d'Unreal Engine.
3. Fermez la fenêtre Visual Studio que vous avez ouverte, car l'éditeur Unreal ouvre une autre fenêtre Visual Studio contenant le projet Unreal et votre projet de jeu.

4. Dans l'éditeur Unreal, effectuez l'une des opérations suivantes :
   - Choisissez un projet Unreal existant que vous souhaitez intégrer à Amazon GameLift.
   - Crée un projet. Pour tester le GameLift plugin Amazon pour Unreal, essayez d'utiliser le modèle Third Person du moteur Unreal. Pour plus d'informations sur ce modèle, consultez la section Modèle à la troisième personne sur le site Web de documentation d'Unreal Engine.

     Vous pouvez également configurer un nouveau projet avec les paramètres suivants :

     - C++
     - Avec du contenu de démarrage
     - Desktop
     - Un nom de projet. Dans les exemples présentés dans cette rubrique, nous avons nommé notre projet`GameLiftUnrealApp`.
5. Dans l'explorateur de solutions de Visual Studio, accédez à l'emplacement de votre projet Unreal. Dans le `Source` dossier Unreal, recherchez un fichier nommé*`Your-application-name`*`.Target.cs`.

   Par exemple : `GameLiftUnrealApp.Target.cs`.
6. Faites une copie de ce fichier et nommez-la :*`Your-application-name`*`Server.Target.cs`.
7. Ouvrez le nouveau fichier et apportez les modifications suivantes :
   - Modifiez le `class` et `constructor` pour qu'il corresponde au nom du fichier.
   - Changez la `Type` forme de `TargetType.Game` à`TargetType.Server`.
   - Le fichier final ressemblera à l'exemple suivant :

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

Votre projet est désormais configuré pour accepter le plug-in SDK GameLift du serveur Amazon.

La tâche suivante consiste à créer les bibliothèques du SDK du serveur C++ pour Unreal afin de pouvoir les importer dans votre projet.

Pour créer les bibliothèques du SDK du serveur C++ pour Unreal

1. Téléchargez le plugin SDK du serveur Amazon GameLift C++ pour Unreal.

   > **Note**
   >
   > Le fait de placer le SDK dans le répertoire de téléchargement par défaut peut entraîner un échec de compilation car le chemin dépasse la limite de 260 caractères. Par exemple : `C:\Users\Administrator\Downloads\GameLift-SDK-Release-06_15_2023\GameLift-Cpp-ServerSDK-5.0.4`
   > Nous vous recommandons de déplacer le SDK vers un autre répertoire, par exemple`C:\GameLift-Cpp-ServerSDK-5.0.4`.

2. Téléchargez et installez OpenSSL. Pour plus d'informations sur le téléchargement d'OpenSSL, consultez la documentation de compilation et d'installation d'OpenSSL sur Github.

   Pour plus d'informations, consultez la documentation OpenSSL Notes pour les plateformes Windows.

   > **Note**
   >
   > La version d'OpenSSL que vous utilisez pour créer le SDK du serveur GameLift Amazon doit correspondre à la version d'OpenSSL utilisée par Unreal pour empaqueter votre serveur de jeu. Vous trouverez les informations de version dans le répertoire `...Engine\Source\ThirdParty\OpenSSL` d'installation d'Unreal.

3. Une fois les bibliothèques téléchargées, créez les bibliothèques du SDK du serveur C++ pour Unreal Engine.

   Dans le `GameLift-Cpp-ServerSDK-<version>` répertoire du SDK téléchargé, compilez avec le `-DBUILD_FOR_UNREAL=1` paramètre et créez le SDK du serveur. Les exemples suivants montrent comment compiler à l'aide de`cmake`.

   Exécutez les commandes suivantes dans votre terminal :

   ```
   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
```

La version Windows crée les fichiers binaires suivants dans le `out\gamelift-server-sdk\Release` dossier :

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

Copiez les deux fichiers de bibliothèque dans le `ThirdParty\GameLiftServerSDK\Win64` dossier du package du plugin Amazon GameLift Unreal Engine.

Utilisez la procédure suivante pour importer le GameLift plugin Amazon dans votre exemple de projet.

Importer le GameLift plugin Amazon

1. Localisez le `GameLiftServerSDK` dossier que vous avez extrait du plugin dans la procédure précédente.
2. Repérez-le `Plugins` dans le dossier racine de votre projet de jeu. (Si le dossier n'existe pas, créez-le ici.)
3. Copiez le `GameLiftServerSDK` dossier dans le`Plugins`.

   Cela permettra au projet Unreal de voir le plugin.
4. Ajoutez le plug-in SDK GameLift du serveur Amazon au `.uproject` fichier du jeu.

   Dans l'exemple, l'application est appelée`GameLiftUnrealApp`, le fichier sera donc appelé`GameLiftUnrealApp.uproject`.
5. Modifiez le `.uproject` fichier pour ajouter le plugin à votre projet de jeu.

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

6. Assurez-vous que le ModuleRules jeu dépend du plugin. Ouvrez le `.Build.cs` fichier et ajoutez la dépendance du GameLiftServer SDK Amazon. Ce fichier se trouve sous*Your-application-name*/Source//*Your-application-name/*.

   Par exemple, le chemin du fichier du didacticiel est. `../GameLiftUnrealApp/Source/GameLiftUnrealApp/GameLiftUnrealApp.Build.cs`

7. Ajouter `"GameLiftServerSDK"` à la fin de la liste des`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;
    }
  }
```

Le plugin devrait maintenant fonctionner pour votre application. Passez à la section suivante pour intégrer les GameLift fonctionnalités Amazon à votre jeu.

Ajoutez le code GameLift du serveur Amazon à votre projet Unreal

Vous avez configuré et configuré votre environnement Unreal Engine, et vous pouvez désormais intégrer un serveur de jeu à Amazon GameLift. Le code présenté dans cette rubrique permet d'effectuer les appels requis vers le GameLift service Amazon. Il implémente également un ensemble de fonctions de rappel qui répondent aux demandes du GameLift service Amazon. Pour plus d'informations sur chaque fonction et sur le rôle du code, voir Initialiser le processus du serveur. Pour plus d'informations sur les actions du SDK et les types de données utilisés dans ce code, consultez. Référence GameLift du SDK Amazon Server pour Unreal Engine

Pour initialiser un serveur de jeu avec Amazon GameLift, suivez la procédure ci-dessous.

> **Note**
>
> Le code GameLift spécifique à Amazon fourni dans la section suivante dépend de l'utilisation d'un indicateur de `WITH_GAMELIFT` préprocesseur. Cet indicateur n'est vrai que lorsque les deux conditions suivantes sont remplies :
>
> - `Target.Type == TargetRules.TargetType.Server`
> - Les plugins ont détecté les fichiers binaires GameLift du SDK du serveur Amazon.
>
> Cela garantit que seules les versions d'Unreal Server invoquent GameLift l'API principale d'Amazon. Il vous permet également d'écrire du code qui s'exécutera correctement pour toutes les cibles Unreal que votre jeu pourrait produire.

Intégrer un serveur de jeu à Amazon GameLift

1. Dans Visual Studio, ouvrez le `.sln` fichier correspondant à votre application. Dans notre exemple, le fichier se `GameLiftUnrealApp.sln` trouve dans le dossier racine.
2. La solution étant ouverte, localisez le *`Your-application-name`*`GameMode.h` fichier de votre application. Exemple: `GameLiftUnrealAppGameMode.h`.
3. Modifiez le fichier d'en-tête pour l'aligner sur l'exemple de code suivant. Assurez-vous de remplacer « GameLiftUnrealApp » par le nom de votre propre application.

```
#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. Ouvrez le *Your-application-name*GameMode.cpp fichier source correspondant. Dans notre exemple :GameLiftUnrealAppGameMode.cpp. et modifiez le code pour l'aligner sur l'exemple de code suivant. Assurez-vous de remplacer « GameLiftUnrealApp » par le nom de votre propre application.

   Cet exemple montre comment ajouter tous les éléments nécessaires à l'intégration avec Amazon GameLift, comme décrit dans Ajouter Amazon GameLift à votre serveur de jeu. Cela consiste notamment à :

   - Initialisation d'un client d' GameLift API Amazon.
   - Implémentation de fonctions de rappel pour répondre aux demandes du GameLift service Amazon, notamment OnStartGameSessionOnProcessTerminate, etonHealthCheck.
   - Appelez ProcessReady () avec un port désigné pour avertir Amazon GameLiftservice lorsqu'il est prêt à héberger des sessions de jeu.

```
#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. Créez un projet de jeu pour les deux types de cibles suivants : éditeur de développement et serveur de développement.

> **Note**
>
> Il n'est pas nécessaire de reconstruire la solution. Créez plutôt uniquement le projet dans le `Games` dossier correspondant au nom de votre application. Sinon, Visual Studio reconstruit l'intégralité du projet UE5, ce qui peut prendre jusqu'à une heure.

6. Une fois les deux versions terminées, fermez Visual Studio et ouvrez le `.uproject` fichier de votre projet pour l'ouvrir dans l'éditeur Unreal.
7. Dans Unreal Editor, empaquetez la version du serveur de votre jeu. Pour choisir une cible, allez dans Plateformes, Windows et sélectionnez ***Y our-application-name Server***.
8. Pour démarrer le processus de création de l'application serveur, accédez à Platforms, Windows et sélectionnez Package Project. Lorsque le build est terminé, vous devriez avoir un exécutable. Dans le cas de notre exemple, le nom du fichier est`GameLiftUnrealAppServer.exe`.
9. La création d'une application serveur dans Unreal Editor produit deux exécutables. L'un d'eux se trouve à la racine du dossier de compilation du jeu et sert de wrapper à l'exécutable du serveur lui-même.

   Lorsque vous créez une GameLift flotte Amazon avec la version de votre serveur, nous vous recommandons de transmettre le fichier exécutable du serveur lui-même comme chemin de lancement de la configuration d'exécution. Par exemple, dans le dossier de compilation de votre jeu, vous pouvez avoir un `GameLiftFPS.exe` fichier à la racine et un autre à`\GameLiftFPS\Binaries\Win64\GameLiftFPSServer.exe`. Lorsque vous créez une flotte, nous vous recommandons de l'utiliser `C:\GameLiftFPS\Binaries\Win64\GameLiftFPSServer.exe` comme chemin de lancement de la configuration d'exécution.
10. Assurez-vous d'ouvrir les ports UDP nécessaires sur la GameLift flotte Amazon, afin que le serveur de jeu puisse communiquer avec les clients du jeu. Par défaut, Unreal Engine utilise le port7777. Pour plus d'informations, consultez UpdateFleetPortSettingsle guide de référence GameLift de l'API de service Amazon.
11. Créez un `install.bat` fichier pour le build de votre jeu. Ce script d'installation s'exécute chaque fois que la version du jeu est déployée sur une GameLift flotte Amazon. Voici un exemple de `install.bat` fichier :

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

Pour certaines versions d'Unreal Engine, le `install.bat` devrait plutôt être

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

> **Note**
>
> Le chemin d'accès au `<>PrereqSetup_x64.exe` fichier est`Engine\Extras\Redist\en-us`.

12. Vous pouvez désormais empaqueter et télécharger votre build de jeu sur Amazon GameLift.

    La version d'OpenSSL que vous empaqueter avec votre build de jeu doit correspondre à la version utilisée par le moteur de jeu lors de la création du serveur de jeu. Assurez-vous d'intégrer la bonne version d'OpenSSL à la version de votre serveur de jeu. Pour le système d'exploitation Windows, le format OpenSSL est. `.dll`

    > **Note**
    >
    > Package les DLL OpenSSL dans la version de votre serveur de jeu. Veillez à empaqueter la même version d'OpenSSL que celle que vous avez utilisée lors de la création du serveur de jeu.
    >
    > - `libssl-1_1-x64.dll`
    >
    >   `libcrypto-1_1-x64.dll`
    >
    > Package vos dépendances ainsi que le fichier exécutable de votre serveur de jeu à la racine d'un fichier zip. Par exemple, `openssl-lib` les dll doivent se trouver dans le même répertoire que le `.exe` fichier.

Étapes suivantes

Vous avez configuré et configuré votre environnement Unreal Engine, et vous pouvez maintenant commencer à GameLift intégrer Amazon dans votre jeu.

Pour plus d'informations sur l'ajout GameLift d'Amazon à votre jeu, consultez ce qui suit :

- Ajoutez Amazon GameLift à votre serveur de jeu
- Référence GameLift du SDK Amazon Server pour Unreal Engine

Pour obtenir des instructions sur le test de votre jeu, consultez Testez votre intégration à l'aide des GameLift Anywhere flottes Amazon.

Intégrer Amazon GameLift dans un projet Unity

Cette rubrique explique comment configurer le plugin SDK Amazon GameLift C# Server pour Unity et l'intégrer dans vos projets de jeu.

Ressources supplémentaires :

- Site de téléchargement GameLift du SDK Amazon Server
- Référence GameLift du SDK Amazon Server 5.x pour C# et Unity
- the section called “Soutien au développement”

Prérequis

Pour utiliser le plug-in SDK du serveur Amazon GameLift C# pour Unity, vous avez besoin des composants suivants :

- Un environnement de développement et une version de Unity Editor compatibles avec le plugin (voirAssistance au développement avec Amazon GameLift). Pour plus d'informations sur les versions de Unity, consultez la section Configuration requise pour Unity dans la documentation Unity.
- Le plug-in SDK GameLift du serveur Amazon pour le package Unity. Ce package inclut le SDK du serveur 5+ pour C#. Vous pouvez télécharger le package à partir de ce site : Getting Started with Amazon GameLift.
- Le registre UnityNuGet délimité par des tiers. Cet outil gère les DLL tierces. Pour plus d'informations, consultez le référentiel UnityNuGetGithub.

Configurer UnityNuGet

Si vous n'avez pas UnityNuGet configuré votre projet de jeu, procédez comme suit pour installer l'outil à l'aide du gestionnaire de packages Unity. Vous pouvez également utiliser la NuGet CLI pour

télécharger manuellement les DLL. Pour plus d'informations, consultez le SDK du serveur Amazon GameLift C# pour Unity. `README`

À UnityNuGet intégrer à votre projet de jeu

1. Votre projet étant ouvert dans l'éditeur Unity, accédez au menu principal et sélectionnez Modifier, Paramètres du projet. Parmi les options, choisissez la section Package Manager et ouvrez le groupe Scoped Registries.
2. Cliquez sur le bouton + et entrez les valeurs suivantes pour le registre UnityNuGet délimité :

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

3. Pour les utilisateurs de la version Unity 2021 :

   Après la configuration UnityNuGet, vérifiez qu'aucune `Assembly Version Validation` erreur ne s'affiche dans la console Unity. Ces erreurs se produisent si les redirections de liaison pour les assemblages aux noms forts dans les NuGet packages ne sont pas résolues correctement vers les chemins de votre projet Unity. Pour résoudre ce problème, configurez la validation de la version d'assemblage de Unity :

   a. Dans l'éditeur Unity, accédez au menu principal, sélectionnez Modifier, Paramètres du projet, puis ouvrez la section Lecteur.

   b. Désélectionnez l'option Validation de la version d'assemblage.

Installez le plugin

Utilisez la procédure suivante pour installer le plug-in SDK du serveur Amazon GameLift C# pour Unity et configurer la journalisation log4net.

Pour installer le plug-in

1. Votre projet étant ouvert dans l'éditeur Unity, accédez au menu principal et sélectionnez Window, Package Manager.
2. Cliquez sur le bouton + pour ajouter un nouveau package. Choisissez l'option Ajouter un package à partir de l'archive tar.

3. Dans Sélectionner les packages sur disque, recherchez le plug-in du SDK Amazon GameLift C# Server pour les fichiers de téléchargement Unity, puis choisissez le fichier du SDK `.tgz` Amazon GameLift Server. Choisissez Ouvrir pour installer le plugin.

Le SDK GameLift du serveur Amazon utilise le framework log4net pour générer des messages de journal. Il est configuré pour envoyer des messages au terminal d'une version de serveur par défaut, mais Unity nécessite une configuration pour ajouter le support de journalisation des fichiers. Vous pouvez ajouter ce support à votre projet en important l'exemple fourni dans le package SDK Amazon GameLift Server. Utilisez la procédure suivante pour ajouter l'exemple et configurer log4net :

Pour configurer log4net pour la sortie de fichiers

1. Votre projet étant ouvert dans l'éditeur Unity, accédez au menu principal et sélectionnez Window, Package Manager.
2. Dans le menu déroulant, sélectionnez Packages : In Project, puis sélectionnez Amazon GameLift Server SDK dans la liste des packages. Cela ouvre les détails du package.
3. Dans les détails du package, sélectionnez l'option Groupe d'échantillons et appuyez sur Importer.
4. Le `log4net.config` fichier et le `LoggingConfiguration.cs` script qui l'accompagne exécutent automatiquement la configuration, qui est désormais configurée dans le `Assets/Samples` dossier du projet.

> **Note**
>
> Si vous devez déplacer votre `log4net.config` fichier vers un autre dossier du projet, vous devez également mettre à jour le chemin du fichier de configuration dans le script `LoggingConfiguration.cs` avec le nouveau chemin. Pour plus d'informations, consultez le manuel de log4net sur la configuration de log4net.

Pour des instructions plus détaillées et des conseils de test, consultez le `README` lien situé dans le téléchargement du plugin.

Configurez une GameLift Anywhere flotte Amazon à des fins de test

Vous pouvez configurer votre station de développement en tant que parc d' GameLift Anywherehébergement Amazon pour tester de manière itérative votre GameLift intégration Amazon.

Avec cette configuration, vous pouvez démarrer les processus du serveur de jeu sur votre poste de travail, envoyer des demandes d'adhésion de joueurs ou de matchmaking GameLift à Amazon pour démarrer des sessions de jeu, et connecter les clients aux nouvelles sessions de jeu. Avec votre propre poste de travail configuré en tant que serveur d'hébergement, vous pouvez surveiller tous les aspects de l'intégration de vos jeux avec Amazon GameLift.

Pour obtenir des instructions sur la configuration de votre poste de travail, reportez-vous Testez votre intégration à l'aide des GameLift Anywhere flottes Amazon à la section pour effectuer les étapes suivantes :

1. Créez un emplacement personnalisé pour votre poste de travail.
2. Créez une GameLift Anywhere flotte Amazon avec votre nouvel emplacement personnalisé. En cas de succès, cette demande renvoie un numéro de flotte. Notez cette valeur, car vous en aurez besoin plus tard.
3. Enregistrez votre station de travail en tant qu'ordinateur dans le nouveau Anywhere parc. Entrez un nom de calcul unique et spécifiez l'adresse IP de votre station de travail. En cas de succès, cette demande renvoie un point de terminaison du SDK de service, sous la forme d'une WebSocket URL. Notez cette valeur, car vous en aurez besoin plus tard.
4. Générez un jeton d'authentification pour le calcul de votre station de travail. Cette authentification de courte durée inclut le jeton et une date d'expiration. Votre serveur de jeu l'utilise pour authentifier les communications avec le GameLift service Amazon. Stockez l'authentification sur l'ordinateur de votre poste de travail afin que les processus de votre serveur de jeu en cours puissent y accéder.

Ajoutez le code GameLift du serveur Amazon à votre projet Unity

Votre serveur de jeu communique avec le GameLift service Amazon pour recevoir des instructions et signaler le statut en cours. Pour ce faire, vous devez ajouter du code de serveur de jeu qui utilise le SDK GameLift du serveur Amazon.

L'exemple de code fourni illustre les éléments d'intégration de base requis. Il utilise un `MonoBehavior` pour illustrer une simple initialisation d'un serveur de jeu avec Amazon GameLift. L'exemple suppose que le serveur de jeu fonctionne sur une GameLift Anywhere flotte Amazon à des fins de test. Il inclut un code pour :

- Initialisez un client d' GameLift API Amazon. L'exemple utilise la version de `InitSDK()` avec les paramètres du serveur pour votre Anywhere flotte et vos calculs. Utilisez l' WebSocket URL, l'ID

de flotte, le nom de calcul (ID d'hôte) et le jeton d'authentification, tels que définis dans la rubrique précédenteConfigurez une GameLift Anywhere flotte Amazon à des fins de test.

- Implémentez des fonctions de rappel pour répondre aux demandes du GameLift service Amazon, notamment `OnStartGameSession``OnProcessTerminate`, et`onHealthCheck`.
- Appelez ProcessReady () avec un port désigné pour informer le GameLift service Amazon lorsque le processus est prêt à héberger des sessions de jeu.

Le code présenté dans cette rubrique établit la communication avec le GameLift service Amazon et. Il implémente également un ensemble de fonctions de rappel qui répondent aux demandes du. Pour plus d'informations sur chaque fonction et sur le rôle du code, voir Initialiser le processus du serveur. Pour plus d'informations sur les actions du SDK et les types de données utilisés dans ce code, consultezRéférence GameLift du SDK Amazon Server pour C#.

Cet exemple montre comment ajouter tous les éléments requis, comme décrit dans Ajouter Amazon GameLift à votre serveur de jeu. Il inclut :

Pour plus d'informations sur l'ajout de GameLift fonctionnalités Amazon, consultez les rubriques suivantes :

- Ajoutez Amazon GameLift à votre serveur de jeu
- Référence GameLift du SDK Amazon Server pour 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);
        }
    }
}
```

Ressources supplémentaires

Utilisez les ressources suivantes pour tester votre serveur de jeu et étendre ses fonctionnalités :

- Configurez votre machine de développement en tant que flotte Amazon GameLift Anywhere et utilisez-la pour des tests locaux. Voir Tester l'intégration personnalisée de votre serveur.
- Créez votre serveur de jeu et téléchargez le build sur Amazon GameLift. Consultez la section Importer une version de serveur personnalisée sur Amazon GameLift.
- Déployez la version de votre serveur de jeu sur une flotte EC2 GameLift gérée par Amazon. Consultez la section Créer une nouvelle GameLift flotte Amazon.

Testez votre intégration à l'aide des GameLift Anywhere flottes Amazon

Vous pouvez utiliser une GameLift Anywhere flotte Amazon pour créer et tester de manière itérative l'intégration de votre jeu avec AmazonGameLift. Configurez votre propre matériel en tant que Anywhere flotte avec une connexion au GameLift service Amazon, puis installez et exécutez votre serveur de jeu sur celui-ci. Utilisez une application de test pour exécuter des scénarios tels que le démarrage/l'arrêt de sessions de jeu, le suivi des connexions des joueurs et la gestion des remblais de matchmaking. Avec une Anywhere flotte, vous pouvez mettre à jour la version de votre serveur de jeu selon vos besoins et bénéficier d'une visibilité complète sur l'activité d'hébergement.

Vous pouvez utiliser GameLift Anywhere des flottes Amazon avec des jeux intégrés au SDK Amazon GameLift Server version 5 ou ultérieure.

Rubriques

- Développement initial
- Itérez sur votre serveur de jeu

Développement initial

Vous avez développé votre jeu et vous êtes en train de l'intégrer au SDK GameLift du serveur Amazon. Pour tester votre intégration, vous pouvez charger chaque nouvelle version de votre serveur de jeu sur Amazon GameLift et créer une flotte. Par ailleurs, l'utilisation d'une Anywhere flotte avec votre ordinateur portable de développement vous permet de réaliser des développements et des tests itératifs de manière plus efficace.

Utilisez les procédures suivantes pour créer une Anywhere flotte et démarrer une session de jeu sur votre ordinateur portable à l'aide de la GameLift console Amazon ou du AWS Command Line Interface (AWS CLI).

Console

1. Ouvrez la GameLiftconsole Amazon.
2. Dans le volet de navigation, sous Hébergement, choisissez Emplacements.
3. Choisissez Créer un lieu.
4. Dans la boîte de dialogue Créer un emplacement, procédez comme suit :
   a. Entrez un nom de lieu. Cela indique l'emplacement des ressources informatiques qu'Amazon GameLift utilise pour exécuter vos jeux au sein de Anywhere flottes. Les noms de lieux personnalisés doivent commencer par custom-.
   b. Sélectionnez Create (Créer).
5. Pour créer une Anywhere flotte, procédez comme suit :
   a. Dans le volet de navigation, sous Hébergement, choisissez Fleets.
   b. Sur la page Fleets (Flottes), choisissez Create fleet (Créer une flotte).
   c. À l'étape Choisir le type de calcul, choisissez N'importe où, puis cliquez sur Suivant.
   d. À l'étape Définir les détails du parc, définissez votre nouveau parc. Pour plus d'informations, veuillez consulter Créez une nouvelle GameLift flotte Amazon.
   e. À l'étape Sélectionner les emplacements, sélectionnez l'emplacement personnalisé que vous avez créé.
   f. Effectuez les étapes de création de flotte restantes pour créer votre Anywhere flotte.
6. Enregistrez votre ordinateur portable en tant que ressource informatique dans le parc que vous avez créé. Utilisez la `register-compute`commande (ou l'opération de `RegisterCompute`l'API). Incluez ce qui `fleet-id` a été créé à l'étape précédente et ajoutez-en un `compute-name` et celui de votre ordinateur portable`ip-address`.

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

Exemple de sortie :

```
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. Démarrez une session de débogage sur votre serveur de jeu.

   a. Obtenez le jeton d'autorisation pour votre ordinateur portable dans le parc que vous avez créé. Utilisez la `get-compute-auth-token`commande (ou l'opération de `GetComputeAuthToken`l'API).

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

   Exemple de sortie :

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

   b. Lancez une instance de débogage de l'exécutable de votre serveur de jeu. Pour exécuter l'instance de débogage, votre serveur de jeu doit appelerInitSDK(). Une fois que le processus est prêt à héberger une session de jeu, le serveur de jeu appelleProcessReady().

8. Créez une session de jeu pour tester votre première intégration avec Amazon GameLiftAnywhere. Utilisez la `create-game-session`commande (ou l'opération de `CreateGameSession`l'API). Spécifiez l'emplacement personnalisé de la flotte.

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

Exemple de sortie :

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

Amazon GameLift envoie un on`StartGameSession()` message au processus de serveur que vous avez enregistré. Le message contient l'`GameSession`objet de l'étape précédente avec les propriétés du jeu, les données des sessions de jeu, les données du système de matchmaking, et plus d'informations sur la session de jeu.

9. Ajoutez de la logique à votre serveur de jeu afin que le processus de votre serveur réponde au on`StartGameSession()` message par`ActivateGameSession()`. L'opération envoie un accusé de réception à Amazon indiquant GameLift que votre serveur a reçu et accepté le message de création d'une session de jeu. Pour plus d'informations, consultez Référence GameLift du SDK pour serveurs Amazon.

Votre serveur de jeu lance à présent une session de jeu que vous pouvez tester et utiliser pour des itérations. Pour savoir comment itérer sur votre serveur de jeu, passez à la section suivante.

AWS CLI

1. Créez un emplacement personnalisé à l'aide de la `create-location`commande (ou de l'opération `CreateLocation`API). Un emplacement personnalisé indique l'emplacement de votre matériel qu'Amazon GameLift utilise pour exécuter vos jeux en Anywhere flottes.

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

Exemple de sortie :

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

```
    }
}
```

2. Créez une Anywhere flotte avec votre emplacement personnalisé à l'aide de la `create-fleet`commande (ou de l'opération `CreateFleet`API). Amazon GameLift crée la flotte dans votre région d'origine et les emplacements personnalisés que vous fournissez.

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

Exemple de sortie :

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

3. Enregistrez votre ordinateur portable en tant que ressource informatique dans le parc que vous avez créé. Utilisez la `register-compute`commande (ou l'opération de `RegisterCompute`l'API). Incluez ce qui `fleet-id` a été créé à l'étape précédente et ajoutez-y un `compute-name` et le public de votre ordinateur portable`ip-address`.

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

Exemple de sortie :

```
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. Démarrez une session de débogage sur votre serveur de jeu.

   a. Obtenez le jeton d'autorisation pour votre ordinateur portable dans le parc que vous avez créé. Utilisez la `get-compute-auth-token`commande (ou l'opération de `GetComputeAuthToken`l'API).

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

   Exemple de sortie :

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

   b. Lancez une instance de débogage de l'exécutable de votre serveur de jeu. Pour exécuter l'instance de débogage, votre serveur de jeu doit appeler`InitSDK()`. Une fois que le processus est prêt à héberger une session de jeu, le serveur de jeu appelle`ProcessReady()`.

5. Créez une session de jeu pour tester votre première intégration avec Amazon GameLiftAnywhere. Utilisez la `create-game-session`commande (ou l'opération de `CreateGameSession`l'API).

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

   Exemple de sortie :

   ```
   GameSession {
       FleetId = fleet-1234,
       GameSessionId = 1111-1111,
   ```

```
    Name = DebugSession,
    IpAddress = 10.1.2.3,
    Port = 1024,
    ...
}
```

Amazon GameLift envoie un `onStartGameSession()` message au processus de serveur que vous avez enregistré. Le message contient l'`GameSession`objet de l'étape précédente avec les propriétés du jeu, les données des sessions de jeu, les données du système de matchmaking, et plus d'informations sur la session de jeu.

6. Ajoutez de la logique à votre serveur de jeu afin que le processus de votre serveur réponde au `onStartGameSession()` message par`ActivateGameSession()`. L'opération envoie un accusé de réception à Amazon indiquant GameLift que votre serveur a reçu et accepté le message de création d'une session de jeu. Pour plus d'informations, consultez Référence GameLift du SDK pour serveurs Amazon.

Votre serveur de jeu lance à présent une session de jeu que vous pouvez tester et utiliser pour des itérations. Pour savoir comment itérer sur votre serveur de jeu, passez à la section suivante.

Itérez sur votre serveur de jeu

Dans ce cas d'utilisation, imaginez un scénario dans lequel vous avez configuré et testé votre serveur de jeu et découvert un bogue. Avec Amazon GameLiftAnywhere, vous pouvez modifier votre code et éviter la configuration fastidieuse liée à l'utilisation d'un parc Amazon EC2.

1. Nettoyez votre existant`GameSession`, si possible. Si le serveur de jeu tombe en panne ou s'il n'appelle pas`ProcessEnding()`, Amazon le GameLift nettoie une `GameSession` fois que le serveur de jeu a cessé d'envoyer des bilans de santé.
2. Modifiez le code sur votre serveur de jeu, compilez-le et préparez-vous pour le prochain test.
3. Votre Anywhere parc précédent est toujours actif et votre ordinateur portable est toujours enregistré en tant que ressource informatique dans le parc. Pour recommencer les tests, créez une nouvelle instance de débogage.

    a. Récupérez le jeton d'autorisation pour votre ordinateur portable dans le parc que vous avez créé. Utilisez la `get-compute-auth-token`commande (ou l'opération de `GetComputeAuthToken`l'API).

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

Exemple de sortie :

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

b. Lancez une instance de débogage de l'exécutable de votre serveur de jeu. Pour exécuter l'instance de débogage, votre serveur de jeu doit appeler`InitSDK()`. Une fois que le processus est prêt à héberger une session de jeu, le serveur de jeu appelle`ProcessReady()`.

4. Votre flotte dispose désormais d'un processus de serveur disponible. Créez votre session de jeu et effectuez vos prochains tests. Utilisez la `create-game-session`commande (ou l'opération de `CreateGameSession`l'API).

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

Amazon GameLift envoie un `onStartGameSession()` message au processus de serveur que vous avez enregistré. Le message contient l'`GameSession`objet de l'étape précédente avec les propriétés du jeu, les données de session de jeu, les données du système de matchmaking et plus d'informations sur la session de jeu.

5. Ajoutez de la logique à votre serveur de jeu afin que le processus de votre serveur réponde au `onStartGameSession()` message par`ActivateGameSession()`. L'opération envoie un accusé de réception à Amazon indiquant GameLift que votre serveur a reçu et accepté le message de création d'une session de jeu. Pour plus d'informations, consultez Référence GameLift du SDK pour serveurs Amazon.

Une fois que vous aurez fini de tester votre serveur de jeu, vous pourrez continuer à utiliser Amazon GameLift pour la gestion de votre flotte et de vos serveurs de jeu. Pour plus d'informations, veuillez consulter Créez une GameLift Anywhere flotte Amazon.

Testez votre intégration à l'aide d'Amazon GameLift Local

> **Note**
>
> Utilisez cette procédure de test si vous utilisez une version du SDK du GameLift serveur Amazon qui est la version 4.x ou une version antérieure. Le package SDK de votre serveur inclut une version compatible d'Amazon GameLift Local. Si vous utilisez la version 5.x du SDK Testez votre intégration à l'aide des GameLift Anywhere flottes Amazon pour serveurs, consultez les tests locaux avec une flotte Amazon GameLiftAnywhere.

Utilisez Amazon GameLift Local pour exécuter une version limitée du GameLift service Amazon géré sur un appareil local et tester l'intégration de votre jeu par rapport à celui-ci. Cet outil est utile lors de l'exécution d'un développement itératif sur l'intégration de votre jeu. L'alternative, qui consiste à charger chaque nouvelle version sur Amazon GameLift et à configurer une flotte pour héberger votre jeu, peut prendre 30 minutes ou plus à chaque fois.

Avec Amazon GameLift Local, vous pouvez vérifier les points suivants :

- Votre serveur de jeu est correctement intégré au SDK Server et communique correctement avec le GameLift service Amazon pour démarrer de nouvelles sessions de jeu, accepter de nouveaux joueurs et signaler son état de santé et son état.
- Votre client de jeu est correctement intégré au AWS SDK pour Amazon GameLift et peut récupérer des informations sur les sessions de jeu existantes, démarrer de nouvelles sessions de jeu, associer des joueurs à des jeux et se connecter à la session de jeu.

Amazon GameLift Local est un outil de ligne de commande qui permet de démarrer une version autonome du service Amazon géré. GameLift Amazon GameLift Local fournit également un journal des événements en cours d'exécution concernant l'initialisation des processus du serveur, les contrôles de santé, ainsi que les appels et réponses d'API. Amazon GameLift Local reconnaît un sous-ensemble des actions du AWS SDK pour Amazon. GameLift Les appels peuvent être effectués à partir de l'AWS CLI ou de votre client de jeu. Toutes les actions de l'API s'exécutent localement comme elles le font dans le service GameLift Web Amazon.

Chaque processus du serveur ne doit héberger qu'une seule session de jeu. La session de jeu est l'exécutable que vous utilisez pour vous connecter à Amazon GameLift Local. Lorsque la session de jeu est terminée, vous devez appeler `GameLiftServerSDK::ProcessEndning` puis quitter le processus. Lorsque vous testez localement avec Amazon GameLift Local, vous pouvez démarrer plusieurs processus de serveur. Chaque processus se connectera à Amazon GameLift Local. Vous pouvez ensuite créer une session de jeu pour chaque processus du serveur. À la fin de votre session de jeu, le processus de votre serveur de jeu doit s'arrêter. Vous devez ensuite démarrer manuellement un autre processus serveur.

Amazon GameLift local prend en charge les API suivantes :

- CreateGameSession
- CreatePlayerSession
- CreatePlayerSessions
- DescribeGameSessions
- DescribePlayerSessions

Configurer Amazon GameLift local

Amazon GameLift Local est fourni sous la forme d'un `.jar` fichier exécutable intégré au SDK du serveur. Il peut être exécuté sous Windows ou Linux et utilisé avec n'importe quelle langue GameLift prise en charge par Amazon.

Avant d'exécuter Local, les éléments suivants doivent avoir été installés.

- Une version du SDK Amazon GameLift Server, versions 3.1.5 à 4.x.
- Java 8

Tester un serveur de jeu

Si vous souhaitez uniquement tester votre serveur de jeu, vous pouvez utiliser le AWS CLI pour simuler les appels du client de jeu vers le service Amazon GameLift Local. Vous pouvez ainsi vérifier que votre serveur de jeux se comporte comme prévu dans les circonstances suivantes :

- Le serveur de jeu démarre correctement et initialise le SDK Amazon GameLift Server.
- Dans le cadre du processus de lancement, le serveur de jeu informe Amazon GameLift qu'il est prêt à héberger des sessions de jeu.

- Le serveur de jeu envoie l'état de santé à Amazon GameLift toutes les minutes pendant qu'il fonctionne.
- Le serveur de jeux répond aux demandes de démarrage d'une nouvelle session de jeu.

1. Démarrez Amazon GameLift Local.

   Ouvrez une fenêtre d'invite de commande, accédez au répertoire contenant le fichier *GameLiftLocal.jar* et exécutez-le. Par défaut, Local écoute les demandes des clients de jeu sur le port 8080. Pour spécifier un autre numéro de port, utilisez le paramètre `-p`, comme indiqué dans l'exemple suivant:

   ```
   java -jar GameLiftLocal.jar -p 9080
   ```

   Une fois que Local a démarré, les journaux indiquent que deux serveurs locaux ont été lancés, l'un qui écoute votre serveur de jeux et l'autre qui écoute votre client de jeu ou l'AWS CLI. Les journaux continuent de signaler l'activité sur les deux serveurs locaux, y compris la communication vers et depuis les composants de votre jeu.

2. Démarrez votre serveur de jeux.

   Démarrez votre serveur GameLift de jeu intégré à Amazon localement. Il n'est pas nécessaire de modifier le point de terminaison du serveur de jeux.

   Dans la fenêtre d'invite de commande locale, les messages du journal indiquent que votre serveur de jeu s'est connecté au service Amazon GameLift Local. Cela signifie que votre serveur de jeu a correctement initialisé le SDK Amazon GameLift Server (avec`InitSDK()`). Il a appelé `ProcessReady()` avec les chemins d'accès aux journaux affichés et, en cas de succès, est prêt à héberger une session de jeu. Pendant que le serveur de jeu fonctionne, Amazon GameLift enregistre chaque rapport d'état de santé du serveur de jeu. L'exemple de message de journal suivant illustre l'intégration réussie d'un serveur de jeux :

   ```
   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
```

Les messages éventuels d'erreur ou d'avertissement sont les suivants :

- Erreur : « ProcessReady Impossible de trouver un processus avec PiD : <process ID>! InitSDK () a-t-il été invoqué ? »
- Avertissement : « L'état du processus existe déjà pour le processus avec PiD : <process ID>! Est-ce que ProcessReady (...) est invoqué plus d'une fois ? »

3. Démarrez l'AWS CLI.

   Une fois que votre serveur de jeux a appelé `ProcessReady()` avec succès, vous pouvez démarrer les appels clients. Ouvrez une autre fenêtre d'invite de commande et démarrez l'AWS CLI. AWS CLIPar défaut, il utilise le point de terminaison du service GameLift Web Amazon. Vous devez remplacer celui-ci par le point de terminaison Local dans toutes les demandes à l'aide du paramètre `--endpoint-url`, comme indiqué dans l'exemple de demande suivant.

   ```
   AWS gamelift describe-game-sessions --endpoint-url http://localhost:9080  --fleet-
   id fleet-123
   ```

   Dans la AWS CLI fenêtre d'invite de `AWS  gamelift` commandes, les commandes génèrent des réponses, comme indiqué dans la Référence des AWS CLI commandes.

4. Créez une session de jeu.

   À l'aide duAWS CLI, soumettez une CreateGameSession() demande. La demande doit respecter la syntaxe attendue. Pour Local, le paramètre `FleetId` peut être défini avec toute chaîne valide (`^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
   ```

Dans la fenêtre d'invite de commande locale, les messages du journal indiquent qu'Amazon GameLift Local a envoyé un onStartGameSession rappel à votre serveur de jeu. Si une session de jeu a été créée avec succès, votre serveur de jeux répond en appelant 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"
```

Dans la AWS CLI fenêtre, Amazon GameLift répond avec un objet de session de jeu incluant un identifiant de session de jeu. Notez que le statut de la nouvelle session de jeu est Activating. Le statut passe à Actif une fois que votre serveur de jeu l'appelle. ActivateGameSession Si vous souhaitez voir le statut modifié, utilisez l'AWS CLI pour appeler 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
    }
```

```
}
```

Tester un serveur de jeu et un client

Pour vérifier l'intégration complète de votre jeu, y compris la connexion de joueurs à des parties, vous pouvez exécuter votre serveur de jeux et votre client de jeu localement. Cela vous permet de tester les appels programmatiques depuis votre client de jeu vers Amazon GameLift Local. Vous pouvez vérifier les actions suivantes :

- Le client du jeu envoie avec succès des demandes de AWS SDK au service Amazon GameLift Local, notamment pour créer des sessions de jeu, récupérer des informations sur des sessions de jeu existantes et créer des sessions pour les joueurs.
- Le serveur de jeux valide correctement les joueurs lorsqu'ils tentent de rejoindre une session de jeu. Pour les joueurs validés, le serveur de jeux peut récupérer les données des joueurs (si la fonction a été mise en œuvre).
- Le serveur de jeux signale l'abandon d'une connexion quand un joueur quitte le jeu.
- Le serveur de jeux rapporte la fin d'une session de jeu.

1. Démarrez Amazon GameLift Local.

   Ouvrez une fenêtre d'invite de commande, accédez au répertoire contenant le fichier *GameLiftLocal.jar* et exécutez-le. Par défaut, Local écoute les demandes des clients de jeu sur le port 8080. Pour spécifier un autre numéro de port, utilisez le paramètre `-p`, comme indiqué dans l'exemple suivant.

   ```
   ./gamelift-local -p 9080
   ```

   Une fois que Local a démarré, vous voyez que les journaux affichent le lancement de deux serveurs locaux, l'un qui écoute votre serveur de jeux et l'autre qui écoute votre client de jeu ou l'AWS CLI.

2. Démarrez votre serveur de jeux.

   Démarrez votre serveur GameLift de jeu intégré à Amazon localement. Pour plus d'informations sur les journaux de messages, consultez Tester un serveur de jeu.

3. Configurez votre client de jeu pour Local et démarrez-le.

Pour utiliser votre client de jeu avec le service Amazon GameLift Local, vous devez apporter les modifications suivantes à la configuration de votre client de jeu, comme décrit dans Configurer Amazon GameLift sur un service de backend :

- Modifiez l'objet `ClientConfiguration` afin de pointer vers votre point de terminaison Local, tel que `http://localhost:9080`.
- Définissez une valeur d'ID de flotte cible. Pour Local, vous n'avez pas besoin d'un ID de flotte réel ; définissez la flotte cible avec une chaîne valide de votre choix (`^fleet-\S+`), comme `fleet-1a2b3c4d-5e6f-7a8b-9c0d-1e2f3a4b5c6d`.
- Définissez les informations d'identification AWS. Pour Local, vous n'avez pas besoin d'informations d'identification AWS réelles ; vous pouvez définir la clé d'accès et la clé secrète avec toute chaîne de votre choix.

Dans la fenêtre d'invite de commande locale, une fois que vous avez démarré le client du jeu, les messages du journal doivent indiquer qu'il a initialisé le GameLift service Amazon `GameLiftClient` et qu'il a bien communiqué avec lui.

4. Testez les appels du client du jeu vers le GameLift service Amazon.

   Vérifiez que votre client de jeu effectue correctement tout ou partie des appels d'API suivants :

   - CreateGameSession()
   - DescribeGameSessions()
   - CreatePlayerSession()
   - CreatePlayerSessions()
   - DescribePlayerSessions()

   Dans la fenêtre d'invite de commande Local, seuls les appels à `CreateGameSession()` se traduisent par des messages de journaux. Les messages du journal s'affichent lorsqu'Amazon GameLift Local invite votre serveur de jeu à démarrer une session de jeu (`onStartGameSession`rappel) et obtient un message de succès `ActivateGameSession` lorsque votre serveur de jeu l'appelle. Dans la fenêtre de l'AWS CLI, tous les appels d'API entraînent des réponses ou des messages d'erreur tels que documentés.

5. Vérifiez que votre serveur de jeux valide les nouvelles connexions de joueur.

Après avoir créé une session de jeu et une session joueur, établissez une connexion directe à la session de jeu.

Dans la fenêtre d'invite de commande Local, les messages de journaux doivent indiquer que le serveur de jeux a envoyé une demande `AcceptPlayerSession()` pour valider la nouvelle connexion de joueur. Si vous utilisez l'AWS CLI pour appeler `DescribePlayerSessions()`, l'état de la session de joueur doit passer d'Instances réservées à Actif.

6. Vérifiez que votre serveur de jeu communique le statut du jeu et du joueur au GameLift service Amazon.

   Pour GameLift qu'Amazon puisse gérer la demande des joueurs et communiquer correctement les statistiques, votre serveur de jeu doit renvoyer différents statuts à AmazonGameLift. Vérifiez que Local enregistre les événements liés aux actions suivantes. Vous pouvez également utiliser l'AWS CLI pour suivre les changements d'état.

   - Un joueur se déconnecte d'une session de jeu : les messages du journal Amazon GameLift Local doivent indiquer que votre serveur de jeu appelle`RemovePlayerSession()`. Un appel de l'AWS CLI à `DescribePlayerSessions()` doit refléter un changement d'état d'`Active` en `Completed`. Vous pouvez également appeler `DescribeGameSessions()` pour vérifier que le nombre de joueurs en cours de la session a diminué d'une unité.
   - Fin de la session de jeu : les messages du journal Amazon GameLift Local doivent indiquer que votre serveur de jeu appelle`TerminateGameSession()`.

     > **Note**
     >
     > Les directives précédentes étaient d'appeler à `TerminateGameSession()` la fin d'une session de jeu. Cette méthode est obsolète avec le SDK Amazon GameLift Server v4.0.1. Consultez Mettre fin à une session de jeu.

   - Le processus du serveur est terminé : les messages du journal Amazon GameLift Local doivent indiquer que votre serveur de jeu appelle`ProcessEnding()`. Un appel de l'AWS CLI à `DescribeGameSessions()` doit refléter un changement d'état d'`Active` en `Terminated` (ou `Terminating`).

Variations avec le local

Lorsque vous utilisez Amazon GameLift Local, gardez à l'esprit les points suivants :

- Contrairement au service GameLift Web Amazon, Local ne suit pas l'état de santé d'un serveur et ne lance pas le on`ProcessTerminate` rappel. Local arrête simplement la journalisation des rapports d'état pour le serveur de jeux.
- Pour les appels vers le kit SDK AWS, les ID de flotte ne sont pas validés et peuvent être n'importe quelle valeur de chaîne qui répond aux exigences du paramètre (`^fleet-\S+`).
- Les ID de session de jeu créés avec Local ont une structure différente. Ils incluent la chaîne `local`, comme illustré ici :

```
arn:aws:gamelift:local::gamesession/fleet-123/gsess-56961f8e-
db9c-4173-97e7-270b82f0daa6
```

Intégration de jeux aux serveurs Amazon GameLift Realtime

Cette rubrique fournit une vue d'ensemble de la solution Amazon gérée GameLift avec serveurs en temps réel. La présentation explique dans quels cas cette solution convient à votre jeu et comment Realtime Servers prend en charge le jeu multijoueur.

Pour obtenir la feuille de route complète qui vous permettra de lancer votre jeu, consultezFeuille de route d'hébergement GameLift géré par Amazon.

> **Tip**
>
> Pour essayer l'hébergement de serveurs de GameLift jeux sur Amazon, consultezDébuter avec Amazon GameLift.

Que sont les serveurs en temps réel ?

Les serveurs en temps réel sont des serveurs de ready-to-go jeu légers qu'Amazon GameLift met à votre disposition pour vos jeux multijoueurs. Les serveurs en temps réel suppriment le processus de développement, de test et de déploiement d'un serveur de jeu personnalisé. Cette solution peut vous aider à minimiser le temps et les efforts nécessaires pour terminer votre jeu.

Fonctions principales

- Pile réseau complète pour l'interaction entre le client et le serveur du jeu
- Fonctionnalité de base du serveur de jeu

- Logique de serveur personnalisable
- Mises à jour en direct des configurations en temps réel et de la logique du serveur
- FlexMatchmatchmaking
- Contrôle flexible des ressources d'hébergement

Configurez des serveurs en temps réel en créant une flotte et en fournissant un script de configuration. Pour plus d'informations sur la création de serveurs en temps réel et sur la préparation de votre client de jeu, consultezPréparez votre serveur en temps réel.

Comment Realtime Servers gère les sessions de jeu

Vous pouvez ajouter une logique personnalisée pour la gestion des sessions de jeu en l'intégrant au script Realtime. Vous pouvez écrire du code pour accéder à des objets spécifiques au serveur, ajouter une logique pilotée par les événements à l'aide de rappels ou ajouter une logique basée sur des scénarios non liés aux événements.

Comment les clients et les serveurs en temps réel interagissent

Au cours d'une session de jeu, les clients du jeu interagissent en envoyant des messages au serveur Realtime via un service principal. Le service principal relaie ensuite les messages entre les clients du jeu pour échanger des informations sur l'activité, l'état du jeu et les données de jeu pertinentes.

En outre, vous pouvez personnaliser la façon dont les clients et les serveurs interagissent en ajoutant une logique de jeu au script en temps réel. Avec une logique de jeu personnalisée, un serveur en temps réel peut implémenter des rappels pour lancer des réponses basées sur des événements.

Protocole de communication

Les serveurs en temps réel et les clients de jeu connectés communiquent via deux canaux : une connexion TCP pour une diffusion fiable et un canal UDP pour une diffusion rapide. Lorsque vous créez des messages, les clients de jeu choisissent le protocole à utiliser en fonction de la nature du message. La livraison de message est définie sur UDP par défaut. Si aucun canal UDP n'est disponible, Amazon GameLift envoie des messages en utilisant le protocole TCP comme solution de secours.

Contenu des messages

Le contenu des messages se compose de deux éléments : une opération de code obligatoire (opCode) et une charge utile facultative. L'OPcode d'un message identifie l'activité d'un joueur ou un

événement de jeu particulier, et la charge utile fournit des données supplémentaires relatives au code d'opération. Ces deux éléments sont définis par le développeur. Votre client de jeu agit en fonction des OPcodes contenus dans les messages qu'il reçoit.

Groupes de joueurs

Les serveurs en temps réel fournissent des fonctionnalités permettant de gérer des groupes de joueurs. Par défaut, Amazon GameLift place tous les joueurs qui se connectent à un jeu dans un groupe « Tous les joueurs ». De plus, les développeurs peuvent configurer des groupes supplémentaires pour leurs jeux et les joueurs peuvent être membres de plusieurs groupes à la fois. Les membres du groupe peuvent envoyer des messages et partager des données de jeu avec tous les joueurs du groupe. Une utilisation possible pour les groupes consiste à configurer les équipes de joueurs et à gérer la communication de l'équipe.

Serveurs en temps réel avec certificats TLS

Avec les serveurs en temps réel, l'authentification des serveurs et le chiffrement des paquets de données sont intégrés au service. Ces fonctionnalités de sécurité sont activées lorsque vous activez la génération de certificats TLS. Lorsqu'un client de jeu essaie de se connecter à un serveur en temps réel, le serveur répond automatiquement avec le certificat TLS, que le client valide. Amazon GameLift gère le chiffrement à l'aide du protocole TLS pour les communications TCP (WebSockets) et du protocole DTLS pour le trafic UDP.

Personnalisation d'un serveur en temps réel

Un serveur en temps réel fonctionne comme un serveur relais sans état. Le serveur en temps réel relaie des paquets de messages et de données de jeu entre les clients de jeu connectés au jeu. Cependant, le serveur en temps réel n'évalue pas les messages, ne traite pas les données et n'exécute aucune logique de jeu. Ainsi utilisé, chaque client de jeu conserve sa propre vision de l'état du jeu et fournit des mises à jour aux autres joueurs via le serveur relais. Chaque client de jeu est responsable de l'intégration de ces mises à jour et de la conciliation de son propre état de jeu.

Vous pouvez personnaliser vos serveurs en les ajoutant à la fonctionnalité de script en temps réel. Grâce à la logique du jeu, par exemple, vous pouvez créer un jeu dynamique avec une vision officielle de l'état du jeu par le serveur.

Amazon GameLift définit un ensemble de rappels côté serveur pour les scripts en temps réel. Implémentez ces rappels pour ajouter des fonctionnalités pilotées par les événements à votre serveur. Par exemple, vous pouvez :

- Authentifier un joueur lorsqu'un client de jeu essaie de se connecter au serveur.
- Validez si un joueur peut rejoindre un groupe sur demande.
- Déterminez à quel moment vous devez transmettre les messages d'un certain joueur ou à un joueur cible, ou effectuez un traitement supplémentaire en réponse.
- Avertissez tous les joueurs lorsqu'un joueur quitte un groupe ou se déconnecte du serveur.
- Affichez le contenu des objets de session de jeu ou des objets de message et utilisez les données.

Déploiement et mise à jour de serveurs en temps réel

L'un des principaux avantages des serveurs en temps réel est la possibilité de mettre à jour vos scripts à tout moment. Lorsque vous mettez à jour un script, Amazon GameLift distribue la nouvelle version à toutes les ressources d'hébergement en quelques minutes. Une fois qu'Amazon aura GameLift déployé le nouveau script, toutes les nouvelles sessions de jeu créées par la suite utiliseront la nouvelle version du script. (Les sessions de jeu existantes continueront d'utiliser la version d'origine.)

Commencez à intégrer votre jeu à des serveurs en temps réel :

- Intégration d'un client de jeu pour les serveurs en temps réel
- Création d'un script en temps réel

Intégration d'un client de jeu pour les serveurs en temps réel

Cette rubrique explique comment préparer votre client de jeu pour pouvoir rejoindre et participer à des sessions de jeu GameLift hébergées sur Amazon.

Il existe deux ensembles de tâches nécessaires pour préparer votre client de jeu :

- Configurer votre client de jeu pour obtenir plus d'informations sur des jeux existants, demander la mise en relation, démarrer de nouvelles sessions de jeu et réserver des emplacements de session de jeu pour un joueur.
- Permettez à votre client de jeu de rejoindre une session de jeu hébergée sur un serveur en temps réel et d'échanger des messages.

Trouvez ou créez des sessions de jeu et des sessions pour les joueurs

Configurez votre client de jeu pour trouver ou démarrer des sessions de jeu, demander des mises en relation FlexMatch et réserver de l'espace pour les joueurs dans un jeu en créant des sessions de joueur. La meilleure pratique consiste à créer un service principal et à l'utiliser pour envoyer des requêtes directes au GameLift service Amazon lorsqu'il est déclenché par une action du client du jeu. Le service principal transmet ensuite les réponses pertinentes au client du jeu.

1. Ajoutez le AWS SDK à votre client de jeu, initialisez un GameLift client Amazon et configurez-le pour utiliser les ressources d'hébergement de vos flottes et de vos files d'attente. Le AWS SDK est disponible en plusieurs langues ; consultez les GameLift kits SDK Pour des services clients personnalisés Amazon.
2. Ajoutez des GameLift fonctionnalités à votre service principal. Pour des instructions plus détaillées, voir Ajoutez Amazon GameLift à votre client de jeu et Ajouter un FlexMatch matchmaking. Une bonne pratique consiste à utiliser des placements de session de jeu pour créer de nouvelles sessions de jeu. Cette méthode vous permet de tirer pleinement parti GameLift de la capacité d'of à placer rapidement et intelligemment de nouvelles sessions de jeu, ainsi qu'à utiliser les données de latence des joueurs pour minimiser le décalage de jeu. Au minimum, votre service principal doit être en mesure de demander de nouvelles sessions de jeu et de gérer les données de session de jeu en réponse. Vous pouvez également ajouter des fonctionnalités pour rechercher et obtenir des informations sur les sessions de jeu existantes, et demander des sessions de joueur qui réservent de façon efficace un emplacement de joueur dans une session de jeu existante.
3. Renvoyer des informations de connexion vers le client de jeu. Le service principal reçoit des objets de session de jeu et de session de joueur en réponse aux demandes adressées au GameLift service Amazon. Ces objets contiennent des informations, notamment les détails de connexion (adresse IP et port) et l'ID de session du joueur, dont le client de jeu a besoin pour se connecter à la session de jeu en cours sur un serveur en temps réel.

Connectez-vous à des jeux sur des serveurs en temps réel

Permettez à votre client de jeu de se connecter directement à une session de jeu hébergée sur un serveur en temps réel et d'échanger des messages avec le serveur et avec d'autres joueurs.

1. Téléchargez le SDK Realtime Client, créez-le et ajoutez-le à votre projet de client de jeu. Consultez le fichier README pour plus d'informations sur les exigences du kit SDK et des instructions sur la façon de créer les bibliothèques client.

2. Appelez Client() avec une configuration client qui spécifie le type de connexion client/serveur à utiliser.

> **Note**
>
> Si vous vous connectez à un serveur en temps réel qui s'exécute sur une flotte sécurisée avec un certificat TLS, vous devez spécifier un type de connexion sécurisé.

3. Ajoutez les fonctionnalités suivantes à votre client de jeu. Consultez la Référence de l'API client (C#) de Realtime Servers pour plus d'informations.
   - Se connecter et se déconnecter d'un jeu
     - Connect()
     - Disconnect()
   - Envoyer des messages à des destinataires cibles
     - SendMessage()
   - Recevoir et traiter des messages
     - OnDataReceived()
   - Rejoindre des groupes et quitter des groupes de joueurs
     - JoinGroup()
     - RequestGroupMembership()
     - LeaveGroup()
4. Configuration des gestionnaires d'événements pour les rappels de client en fonction de vos besoins. Consultez Référence de l'API client (C#) de Realtime Servers : rappels asynchrones.

Lorsque vous travaillez avec des flottes en temps réel pour lesquelles la génération de certificats TLS est activée, le serveur est automatiquement authentifié à l'aide du certificat TLS. Le trafic TCP et UDP est chiffré en transit pour assurer la sécurité de la couche de transport. Le trafic TCP est chiffré avec TLS 1.2 et le trafic UDP avec DTLS 1.2.

Exemples de clients de jeu

Client en temps réel de base (C#)

Cet exemple illustre une intégration de base du client de jeu avec le SDK Realtime Client (C#). Comme indiqué, l'exemple initialise un objet client en temps réel, configure des gestionnaires

d'événements et implémente les rappels côté client, se connecte à un serveur en temps réel, envoie un message et se déconnecte.

```
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);
        }
    }
}
```

Création d'un script en temps réel

Pour utiliser des serveurs en temps réel pour votre jeu, vous devez fournir un script (sous la forme d'un JavaScript code) pour configurer et éventuellement personnaliser un parc de serveurs en temps réel. Cette rubrique décrit les principales étapes de la création d'un script en temps réel. Une fois le script prêt, chargez-le sur le GameLift service Amazon et utilisez-le pour créer une flotte (voir[Importer un script de serveurs en temps réel sur Amazon GameLift](#)).

Pour préparer un script à utiliser avec des serveurs en temps réel, ajoutez les fonctionnalités suivantes à votre script en temps réel.

Gérer le cycle de vie des sessions de jeu (obligatoire)

Au minimum, un script en temps réel doit inclure la `Init()` fonction qui prépare le serveur en temps réel à démarrer une session de jeu. Il est également vivement recommandé de fournir un moyen de mettre fin à des sessions de jeu, afin de garantir que de nouvelles sessions de jeu pourront être encore démarrées sur votre flotte.

La fonction de `Init()` rappel, lorsqu'elle est appelée, reçoit un objet de session en temps réel, qui contient une interface pour le serveur en temps réel. Veuillez consulter Interface de serveurs en temps réel pour plus de détails sur cette interface.

Pour terminer correctement une session de jeu, le script doit également appeler la fonction du `session.processEnding` serveur en temps réel. Cela requiert un mécanisme pour déterminer quand mettre fin à une session. L'exemple de code de script illustre un mécanisme simple qui recherche les connexions de joueur et déclenche l'arrêt de la session de jeu quand aucun joueur ne s'est connecté à la session depuis un temps spécifié.

Les serveurs en temps réel dotés de la configuration la plus élémentaire (initialisation et terminaison du processus serveur) agissent essentiellement comme des serveurs relais sans état. Le serveur en temps réel transmet les messages et les données de jeu entre les clients de jeu connectés au jeu, mais ne prend aucune mesure indépendante pour traiter les données ou exécuter la logique. Vous pouvez également ajouter une logique de jeu, déclenchée par des événements de jeu ou d'autres mécanismes, en fonction des besoins de votre jeu.

Ajouter une logique de jeu côté serveur (facultatif)

Vous pouvez éventuellement ajouter une logique de jeu à votre script en temps réel. Vous pouvez, par exemple, effectuer une partie ou l'ensemble des opérations suivantes. L'exemple de code de script fournit une illustration. Consultez Référence de script Amazon GameLift Realtime Servers.

- Ajoutez une logique pilotée par les événements. Implémentez les fonctions de rappel pour répondre aux événements client-serveur. Veuillez consulter Rappels de script pour les serveurs en temps réel pour obtenir la liste complète des rappels.
- Déclenchez la logique en envoyant des messages au serveur. Créez un ensemble de codes d'opération spéciaux pour les messages envoyés au serveur à partir des clients de jeu et ajoutez des fonctions pour gérer leur réception. Utilisez le rappel `onMessage` et analysez le contenu du message à l'aide de l'interface `gameMessage` (voir gameMessage.opcode).
- Activez la logique du jeu pour accéder à vos autres AWS ressources. Pour plus de détails, consultez Communiquez avec les autres AWS ressources de vos flottes.
- Autorisez la logique du jeu à accéder aux informations relatives à la flotte de l'instance sur laquelle elle s'exécute. Pour plus de détails, consultez Obtenir des données de flotte pour une GameLift instance Amazon.

Exemple de script de serveurs en temps réel

Cet exemple illustre un script de base nécessaire pour déployer des serveurs en temps réel ainsi qu'une certaine logique personnalisée. Il contient la fonction `Init()` requise et utilise une minuterie pour déclencher l'arrêt de la session de jeu en fonction du temps écoulé sans connexion de joueur. Il inclut également des hooks pour une logique personnalisée, y compris certaines implémentations de rappel.

```
// 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
};
```

Intégration de jeux avec le GameLift plugin Amazon pour Unity

Les rubriques de cette section décrivent le GameLift plugin Amazon pour Unity et expliquent comment l'utiliser pour préparer votre projet de jeu multijoueur en vue de son hébergement sur Amazon GameLift. Travaillez entièrement dans votre environnement de développement Unity grâce aux flux de travail guidés du plugin pour répondre aux exigences de base de l'hébergement chez Amazon GameLift.

Amazon GameLift est un service entièrement géré qui permet aux développeurs de jeux de gérer et de faire évoluer des serveurs de jeux dédiés pour les jeux multijoueurs basés sur des sessions. Pour plus d'informations sur l' GameLift hébergement Amazon, consultezComment GameLift fonctionne Amazon.

- Guide du GameLift plugin Amazon pour Unity pour le SDK de serveur 5.x, version 2.0.0, fonctionne avec le SDK 5.x du serveur et supporte Amazon. GameLift Anywhere
- Guide du GameLift plugin Amazon pour Unity pour le SDK de serveur 4.x, version 1.0.0, fonctionne avec le SDK du serveur 4.x ou une version antérieure. Cette version utilise Amazon GameLift Local pour les tests d'intégration.

Guide du GameLift plugin Amazon pour Unity pour le SDK de serveur 5.x

Amazon GameLift fournit des outils pour préparer vos serveurs de jeux multijoueurs à fonctionner avec Amazon GameLift. Le GameLift plugin Amazon pour Unity facilite l'intégration d'Amazon GameLift dans vos projets de jeux Unity, le test de votre intégration avec Amazon GameLift Anywhere et le déploiement des GameLift ressources Amazon pour l'hébergement dans le cloud.

Ce plugin utilise des AWS CloudFormation modèles pour déployer des solutions d'hébergement pour des scénarios de jeu courants. Utilisez ces solutions telles que fournies ou personnalisez-les selon les besoins de vos jeux.

Rubriques

À propos du plugin

Le plugin pour Unity fournit une expérience de démarrage simplifiée pour intégrer et héberger vos jeux multijoueurs Unity avec Amazon GameLift. Vous pouvez tirer parti des fonctionnalités du plugin et des composants prédéfinis pour que vos jeux soient rapidement opérationnels.

Le plugin ajoute des outils et des fonctionnalités à l'éditeur Unity. Utilisez les flux de travail guidés pour GameLift intégrer Amazon à votre projet de jeu, le tester localement, puis déployer le serveur de jeu sur l'hébergement GameLift cloud Amazon.

Utilisez les solutions d'hébergement prédéfinies du plugin pour déployer votre jeu. Configurez une flotte Amazon GameLift Anywhere avec votre poste de travail local comme hôte. Pour l'hébergement dans le cloud, choisissez entre deux scénarios de déploiement courants qui équilibrent de différentes manières la latence des joueurs, la disponibilité des sessions de jeu et les coûts. Un scénario inclut un système de FlexMatch matchmaking simple et un ensemble de règles. Utilisez ces scénarios pour mettre en place une solution d'hébergement de base prête pour la production, puis optimisez-la et personnalisez-la selon vos besoins.

Le plugin inclut les composants suivants :

- Modules de plug-in pour l'éditeur Unity. Lorsque le plugin est installé, un nouvel élément du menu principal vous donne accès aux GameLift fonctionnalités d'Amazon.
- bibliothèques C# pour l'API de GameLift service Amazon avec fonctionnalités côté client.
- bibliothèques C# pour le SDK GameLift du serveur Amazon (version 5.x).
- Découvrez du contenu de jeu, y compris des éléments et des scènes, afin de pouvoir essayer Amazon GameLift même si vous n'avez pas de jeu multijoueur prêt à être compilé.
- Configurations de solution, fournies sous forme de AWS CloudFormation modèles, que le plugin utilise lors du déploiement de votre serveur de jeu sur le cloud à des fins d'hébergement.

Flux de travail du plugin

Les étapes suivantes décrivent une approche typique d'intégration et de déploiement d'un projet de jeu avec le GameLift plugin Amazon pour Unity. Vous devez effectuer ces étapes en utilisant l'éditeur Unity et le code de votre jeu.

1. Créez un profil utilisateur lié à votre AWS compte et fournissant les informations d'accès à un utilisateur valide autorisé à utiliser Amazon GameLift.
2. Ajoutez du code serveur à votre projet de jeu pour établir la communication entre un serveur de jeu actif et le GameLift service with Amazon.
3. Ajoutez un code client à votre projet de jeu qui permet aux clients du jeu d'envoyer des demandes GameLift à Amazon pour démarrer ou rejoindre une session de jeu, puis de se connecter au serveur de jeu.
4. Utilisez le flux de travail Anywhere pour configurer votre station de travail locale en tant qu'hôte Anywhere pour votre serveur de jeu. Lancez votre serveur de jeu et votre client localement, connectez-vous à une session de jeu et testez votre intégration.
5. Utilisez le flux de travail d'hébergement EC2 pour télécharger votre serveur de jeu intégré et déployer une solution d'hébergement cloud. Lorsque votre serveur de jeu est prêt, lancez votre client de jeu localement, connectez-vous à une session de jeu, connectez-vous et jouez au jeu.

Lorsque vous travaillez dans le plugin, vous allez créer et utiliser AWS des ressources. Ces actions peuvent entraîner des frais pour le AWS compte utilisé. Si vous débutezAWS, les actions peuvent être couvertes par le niveau AWS gratuit.

Installez le plugin pour Unity

Cette section explique comment ajouter le plugin à un projet Unity. Une fois le plugin installé, les fonctionnalités du plugin sont disponibles lorsque le projet est ouvert dans l'éditeur Unity.

Avant de commencer

Voici ce dont vous avez besoin pour utiliser le GameLift plugin Amazon pour Unity :

- Unity pour Windows 2022 LTS ou Unity pour macOS
- Téléchargement GameLift du plugin Amazon pour Unity. [Site de téléchargement] Le téléchargement inclut deux packages :
  - Plug-in GameLift autonome Amazon pour Unity
  - SDK du serveur Amazon GameLift C# pour Unity
- Microsoft Visual Studio 2019 ou version ultérieure.
- Un projet de jeu multijoueur avec un code de jeu C#.
- Le registre UnityNuGet délimité par des tiers. Cet outil gère les DLL tierces. Pour plus d'informations, consultez le référentiel UnityNuGetGithub.

Ajoutez le plugin à votre projet de jeu

Effectuez les tâches suivantes en utilisant l'éditeur Unity et les fichiers de votre projet de jeu.

Étape 1 : Ajoutez UnityNuGet à votre projet de jeu

Si vous n'avez pas UnityNuGet configuré votre projet de jeu, procédez comme suit pour installer l'outil à l'aide du gestionnaire de packages Unity. Vous pouvez également utiliser la NuGet CLI pour télécharger manuellement les DLL. Pour plus d'informations, consultez le SDK du serveur Amazon GameLift C# pour Unity. `README`

1. Votre projet étant ouvert dans l'éditeur Unity, accédez au menu principal et sélectionnez Modifier, Paramètres du projet. Parmi les options, choisissez la section Package Manager et ouvrez le groupe Scoped Registries.
2. Cliquez sur le bouton + et entrez les valeurs suivantes pour le registre UnityNuGet délimité :

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

Pour les utilisateurs de la version Unity 2021 :

Après la configuration UnityNuGet, vérifiez qu'aucune `Assembly Version Validation` erreur ne s'affiche dans la console Unity. Ces erreurs se produisent si les redirections de liaison pour les assemblages aux noms forts dans les NuGet packages ne sont pas résolues correctement vers les chemins de votre projet Unity. Pour résoudre ce problème, configurez la validation de la version d'assemblage de Unity :

1. Dans l'éditeur Unity, accédez au menu principal, sélectionnez Modifier, Paramètres du projet, puis ouvrez la section Lecteur.
2. Désélectionnez l'option Validation de la version d'assemblage.

Étape 2 : Ajouter le plugin et les packages SDK du serveur C#

1. Décompressez le GameLift plugin Amazon pour le téléchargement de Unity, qui contient les deux packages.
2. Votre projet étant ouvert dans l'éditeur Unity, allez dans le menu principal et sélectionnez Window, Package Manager.
3. Cliquez sur le bouton + pour ajouter un nouveau package. Choisissez l'option Ajouter un package à partir de l'archive tar.
4. Dans Sélectionner les packages sur disque, recherchez le plugin SDK Amazon GameLift C# Server pour les fichiers de téléchargement Unity, puis choisissez le `com.amazonaws.gameliftserver.sdk-<version>.tgz` fichier. Choisissez Ouvrir pour installer le plugin.
5. Dans Sélectionner les packages sur disque, recherchez le plugin GameLift autonome Amazon pour les fichiers de téléchargement Unity, puis choisissez le fichier`com.amazonaws.gamelift-<version>.tgz`. Choisissez Ouvrir pour installer le plugin.
6. Vérifiez que le plugin autonome est ajouté à votre projet. Retournez dans la fenêtre de l'éditeur Unity. Consultez le menu principal pour voir s'afficher le nouveau bouton GameLift du menu Amazon.

Étape 3 : Importer l'exemple de jeu (facultatif)

Le plugin pour Unity est fourni avec un ensemble d'exemples de ressources de jeu, y compris des scènes, que vous pouvez ajouter à votre projet de jeu. L'importation de l'exemple de jeu vous permet de tester, de créer et de déployer rapidement un jeu multijoueur simple avec Amazon GameLift.

L'exemple de jeu est déjà entièrement intégré aux GameLift SDK Amazon. Vous pouvez donc ignorer les tâches d'intégration et effectuer les tâches de flux de travail restantes.

En utilisant l'exemple de jeu, vous pouvez configurer et rejoindre une flotte Amazon GameLift Anywhere hébergée localement en quelques minutes. Vous pouvez déployer le jeu sur Amazon GameLift et rejoindre un jeu en direct hébergé dans le cloud en moins d'une heure.

Pour importer l'exemple de jeu, procédez comme suit :

1. Votre projet de jeu étant ouvert dans l'éditeur Unity, accédez au GameLift menu Amazon et sélectionnez Sample Game, Importer Sample Game.
2. Une fois les fichiers importés, accédez à nouveau au GameLift menu Amazon et sélectionnez Sample Game, Initialize Settings. Cette étape configure votre projet pour créer le client et le serveur du jeu.

Une fois l'installation terminée, deux nouvelles scènes seront ajoutées à votre projet de jeu. Vous verrez également d'autres actifs du projet, dont un GameLiftClientSettingsactif.

Pour plus de détails sur l'interface utilisateur et le gameplay de l'exemple, consultez le fichier readme de l'exemple de jeu.

Configuration d'un profil AWS utilisateur

Après avoir installé le plugin, configurez un profil et associez-le à un AWS compte utilisateur valide. Vous pouvez gérer plusieurs profils, mais vous ne pouvez avoir qu'un seul profil actif à la fois. Chaque fois que vous travaillez dans le plugin, sélectionnez un profil à utiliser.

La gestion de plusieurs profils vous permet de passer d'un scénario d'hébergement à un autre. Par exemple, vous pouvez configurer des profils avec les mêmes AWS informations d'identification mais avec des AWS régions différentes. Vous pouvez également configurer des profils avec différents AWS comptes ou avec différents utilisateurs/ensembles d'autorisations.

> **Note**
>
> Si vous avez installé la AWS CLI sur votre poste de travail et qu'un profil est déjà configuré, le GameLift plugin Amazon peut le détecter et le répertoriera comme profil existant. Le plugin sélectionne automatiquement n'importe quel profil nommé`[default]`. Vous pouvez utiliser un profil existant ou en créer un nouveau.

Pour configurer votre AWS profil

1. Dans le menu principal de l'éditeur Unity, choisissez Amazon, GameLift puis sélectionnez Définir les profils de AWS compte. Cette action ouvre la fenêtre du plugin. Ouvrez la page Profils AWS utilisateurs.
2. Si le plugin détecte un profil existant, il ne vous sera pas demandé d'en créer un. Sélectionnez un profil existant ou choisissez Ajouter un autre profil pour en créer un nouveau.
3. Si le plugin ne détecte aucun profil existant, il vous invite à en créer un. Vous pouvez créer un nouveau profil à l'aide d'un compte nouveau ou d'un AWS compte existant.

   > **Note**
   >
   > Vous devez utiliser la console AWS de gestion pour créer un nouveau AWS compte et créer ou mettre à jour un utilisateur doté des autorisations appropriées.

   Lorsque vous configurez un profil, vous avez besoin des informations suivantes :

   - Un compte AWS. Si vous devez créer un nouveau AWS compte, suivez les instructions pour le créer. Voir Créer un AWS compte pour plus de détails.
   - Un utilisateur AWS du compte autorisé à utiliser Amazon GameLift et les autres AWS services requis. Consultez Configurez un Compte AWS les instructions relatives à la configuration d'un utilisateur AWS Identity and Access Management (IAM) avec des GameLift autorisations Amazon.
   - Informations d'identification de votre AWS utilisateur. Cet utilisateur a également besoin d'un accès programmatique avec des informations d'identification à long terme. Ces informations d'identification se composent d'un identifiant de clé d'AWSaccès et d'une clé AWS secrète. Consultez Obtenir vos clés d'accès pour plus de détails.
   - Région AWS. Il s'agit d'un emplacement géographique dans lequel vous souhaitez créer vos AWS ressources pour l'hébergement. Pendant le développement, nous vous recommandons d'utiliser une région proche de votre emplacement physique afin de minimiser le temps de latence. Consultez la liste des AWSrégions prises en charge.
4. Lorsque vous avez sélectionné ou créé un profil, vérifiez l'état de démarrage du profil et prenez les mesures nécessaires. Tous les profils doivent être amorcés pour utiliser les fonctionnalités du GameLift plugin Amazon.

Pour démarrer votre profil, procédez comme suit :

Le bootstrapping désigne un compartiment Amazon S3 à utiliser avec le profil utilisateur sélectionné. Amazon S3 est un AWS service de base pour le stockage de données et d'objets. Le bucket utilisé pour stocker les configurations de projet, créer des artefacts et d'autres dépendances. Les buckets ne sont pas partagés entre d'autres profils.

> **Note**
>
> Le bootstrapping crée de nouvelles AWS ressources et peut entraîner des coûts.

1. Lorsque vous consultez vos profils dans la fenêtre du plugin AWSUser Profiles, sélectionnez le profil que vous souhaitez utiliser. Un message d'avertissement s'affiche si le profil n'a pas encore été amorcé.
2. Dans la section Bootstrap your profile, sélectionnez un profil dans la liste déroulante et vérifiez l'état du bootstrap. Si le statut indique qu'aucun bucket n'existe, cliquez sur le bouton Bootstrap profile. Vous pouvez définir un nouveau nom de compartiment pour le nom du compartiment, saisir un compartiment existant auquel vous avez accès ou conserver le nom généré automatiquement.
3. Attendez que le statut du bootstrap passe à « Actif ». Cette opération peut prendre quelques minutes. Lorsque le statut est « Actif », vous pouvez utiliser le profil pour utiliser les fonctionnalités du plugin

Configurez votre jeu pour des tests locaux avec Amazon GameLift Anywhere

Dans ce flux de travail, vous ajoutez du code de jeu client et serveur pour les GameLift fonctionnalités Amazon et vous utilisez le plugin pour désigner votre station de travail locale comme hôte de serveur de jeu de test. Lorsque vous avez terminé les tâches d'intégration, utilisez le plugin pour créer les composants du client et du serveur de votre jeu.

Pour démarrer le flux de travail Amazon GameLift Anywhere, procédez comme suit :

- Dans le menu principal de l'éditeur Unity, choisissez Amazon, GameLift puis Host with Anywhere. Cette action ouvre la page du plugin permettant de configurer votre jeu avec une Anywhere flotte @. La page présente un processus en cinq étapes pour intégrer, créer et lancer les composants de votre jeu.

Définissez votre profil

Choisissez le profil que vous souhaitez utiliser lorsque vous suivez ce flux de travail. Le profil que vous sélectionnez a un impact sur toutes les étapes du flux de travail. Toutes les ressources que vous créez sont associées au AWS compte du profil et sont placées dans la AWS région par défaut du profil. Les autorisations de l'utilisateur du profil déterminent votre accès aux AWS ressources et aux actions.

1. Sélectionnez un profil dans la liste déroulante des profils disponibles. Si vous n'avez pas encore de profil ou si vous souhaitez en créer un nouveau, rendez-vous dans le GameLift menu Amazon et choisissez Set AWS Account Profiles.
2. Si le statut du bootstrap n'est pas « Actif », choisissez le profil Bootstrap et attendez que le statut passe à « Actif ».

Intégrez votre jeu à Amazon GameLift

> **Note**
>
> Si vous avez importé l'exemple de jeu, vous pouvez ignorer cette étape. Les exemples de ressources de jeu contiennent déjà le code serveur et client nécessaire.

Pour cette étape du flux de travail, vous devez mettre à jour le code du client et du serveur dans votre projet de jeu.

- * Les serveurs de jeu doivent être en mesure de communiquer avec le GameLift service Amazon pour recevoir des instructions les invitant à démarrer une session de jeu, fournir des informations de connexion à une session de jeu et signaler le statut.
- Les clients du jeu doivent être en mesure d'obtenir des informations sur les sessions de jeu, de rejoindre ou de démarrer des sessions de jeu, et d'obtenir des informations de connexion pour rejoindre une partie.

Intégrez le code de votre serveur

Si vous utilisez votre propre projet de jeu avec des scènes personnalisées, utilisez l'exemple de code fourni pour ajouter le code de serveur requis à votre projet de jeu :

1. Dans les fichiers de votre projet de jeu, ouvrez le `Assets/Scripts/Server` dossier. S'il n'existe pas, créez-le.
2. Accédez au GitHub repo aws/ amazon-gamelift-plugin-unity et ouvrez le chemin. `Samples~/SampleGame/Assets/Scripts/Server`
3. Localisez le fichier GameLiftServer .cs. et copiez-le dans le dossier Server de votre projet de jeu. Lorsque vous créez un fichier exécutable pour le serveur, utilisez ce fichier comme cible de génération.

L'exemple de code inclut les éléments minimaux requis suivants, qui utilisent le SDK du serveur Amazon GameLift C# (version 5) :

- Initialise un client d' GameLift API Amazon. L'`InitSDK()`appel avec les paramètres du serveur est obligatoire pour une flotte Amazon GameLift Anywhere. Ces paramètres sont automatiquement définis pour être utilisés dans le plugin.
- Implémente les fonctions de rappel requises pour répondre aux demandes du GameLift service Amazon, notamment `OnStartGameSession``OnProcessTerminate`, et`onHealthCheck`.
- Appels `ProcessReady()` avec un port désigné pour informer le GameLift service Amazon lorsque le processus du serveur est prêt à héberger des sessions de jeu.

Si vous souhaitez personnaliser l'exemple de code du serveur, consultez les ressources suivantes :

- Ajoutez Amazon GameLift à votre serveur de jeu
- Référence GameLift du SDK Amazon Server 5.x pour C# et Unity

Intégrez votre code client

Si vous utilisez votre propre projet de jeu avec des scènes personnalisées, vous devez intégrer les fonctionnalités de base dans votre client de jeu. Vous devez également ajouter des éléments d'interface utilisateur afin que les joueurs puissent se connecter et rejoindre une session de jeu. Utilisez les API du GameLift service Amazon (dans le AWS SDK) pour obtenir des informations sur les sessions de jeu, créer de nouvelles sessions de jeu ou rejoindre des sessions de jeu existantes,

Lorsque vous créez un client pour des tests locaux avec une flotte Anywhere, vous pouvez ajouter des appels directs au GameLift service Amazon. Lorsque vous développez votre jeu pour l'hébergement dans le cloud, ou si vous envisagez d'utiliser des flottes Anywhere pour l'hébergement

de production, vous devez créer un service principal côté client pour gérer toutes les communications entre les clients du jeu et le service Amazon. GameLift

Pour GameLift intégrer Amazon dans votre code client, utilisez les ressources suivantes comme guide.

- Intégrez le client à la GameLiftCoreApi classe dans le GitHub dépôt amazon-gamelift-plugin-unity aws/. Cette classe fournit des contrôles pour l'authentification des joueurs et pour la récupération des informations de session de jeu.
- Consultez des exemples d'intégrations de jeux, disponibles dans le GitHub dépôt amazon-gamelift-plugin-unity aws/,. `Samples~/SampleGame/Assets/Scripts/Client/GameLiftClient.cs`
- Suivez les instructions de la section Ajouter Amazon GameLift à votre client de jeu Unity.

Pour les clients de jeu qui se connectent à une flotte Anywhere, votre client de jeu a besoin des informations suivantes. Le plugin met automatiquement à jour votre projet de jeu pour utiliser les ressources que vous avez créées dans le plugin.

- FleetId - L'identifiant unique de votre flotte Anywhere.
- FleetLocation - L'emplacement personnalisé de votre flotte Anywhere.
- AwsRegion - La AWS région dans laquelle votre flotte Anywhere est hébergée. Il s'agit de la région que vous avez définie dans votre profil utilisateur.
- ProfileName - Un profil AWS d'identification sur votre machine locale qui permet d'accéder au AWS SDK pour GameLift. Le client du jeu utilise ces informations d'identification pour authentifier les demandes adressées au GameLift service Amazon.

> **Note**
>
> Le profil d'identification est généré par le plugin et stocké sur la machine locale. Par conséquent, vous devez exécuter le client sur la machine locale (ou sur une machine ayant le même profil).

Connectez-vous à une flotte n'importe où

Au cours de cette étape, vous désignez une flotte Anywhere à utiliser. Une flotte Anywhere définit un ensemble de ressources informatiques, qui peuvent être situées n'importe où, pour l'hébergement de serveurs de jeux.

- Si le AWS compte que vous utilisez actuellement possède des flottes Anywhere existantes, ouvrez le champ déroulant Nom de la flotte et choisissez une flotte. Cette liste déroulante affiche uniquement les flottes Anywhere de la AWS région correspondant au profil utilisateur actuellement actif.
- S'il n'existe aucune flotte existante, ou si vous souhaitez en créer une nouvelle, choisissez Create new Anywhere fleet et saisissez un nom de flotte.

Une fois que vous avez choisi une flotte Anywhere pour votre projet, Amazon GameLift vérifie que l'état de la flotte est actif et affiche l'identifiant de la flotte. Vous pouvez suivre la progression de cette demande dans le journal de sortie de l'éditeur Unity.

Enregistrer un ordinateur

Au cours de cette étape, vous enregistrez votre poste de travail local en tant que ressource de calcul dans le nouveau parc Anywhere.

1. Entrez un nom de calcul pour votre machine locale. Si vous ajoutez plusieurs ordinateurs dans le parc, les noms doivent être uniques.
2. Choisissez Register compute. Vous pouvez suivre la progression de cette demande dans le journal de sortie de l'éditeur Unreal.

Le plugin enregistre votre station de travail locale avec l'adresse IP définie sur localhost (127.0.0.1). Ce paramètre suppose que vous exécuterez le client et le serveur du jeu sur la même machine.

En réponse à cette action, Amazon GameLift vérifie qu'il peut se connecter au calcul et renvoie des informations sur le calcul nouvellement enregistré.

Lancer le jeu

Au cours de cette étape, vous créez les composants de votre jeu et vous les lancez pour jouer au jeu. Réalisez les tâches suivantes :

1. Configurez votre client de jeu. Au cours de cette étape, vous demandez au plugin de mettre à jour une `GameLiftClientSettings` ressource pour votre projet de jeu. Le plugin utilise cet actif pour stocker certaines informations dont votre client de jeu a besoin pour se connecter au GameLift service Amazon.

    a. Si vous n'avez pas importé et initialisé le jeu d'exemple, créez une nouvelle `GameLiftClientSettings` ressource. Dans le menu principal de l'éditeur Unity,

choisissez Assets, Create GameLift, Client Settings. Si vous créez plusieurs copies de GameLiftClientSettings votre projet, le plugin le détecte automatiquement et vous indique quel actif le plugin va mettre à jour.

b. Dans Launch Game, choisissez Configurer le client : appliquer les paramètres n'importe où. Cette action met à jour les paramètres de votre client de jeu pour utiliser la flotte Anywhere que vous venez de configurer.

2. Créez et exécutez votre client de jeu.

   a. Créez un exécutable client en utilisant le processus de construction standard de Unity. Dans Fichier, Paramètres de compilation, basculez la plateforme sur Windows, Mac, Linux. Si vous avez importé l'exemple de jeu et initialisé les paramètres, la liste des builds et la cible des builds sont automatiquement mis à jour.

   b. Lancez une ou plusieurs instances du nouveau fichier exécutable du client de jeu.

3. Lancez un serveur de jeu dans votre flotte Anywhere. Choisissez Serveur : Lancer le serveur dans l'éditeur. Cette tâche démarre un serveur live auquel votre client peut se connecter tant que l'éditeur Unity reste ouvert.

4. Démarrez ou rejoignez une session de jeu. Dans vos instances de client de jeu, utilisez l'interface utilisateur pour associer chaque client à une session de jeu. La manière dont vous procédez dépend de la manière dont vous avez ajouté des fonctionnalités au client.

Si vous utilisez l'exemple de client de jeu, il présente les caractéristiques suivantes :

- Un composant de connexion des joueurs. Lorsque vous vous connectez à un serveur de jeu d'une flotte Anywhere, aucune validation n'est requise par le joueur. Vous pouvez saisir n'importe quelle valeur pour rejoindre la session de jeu.
- Une interface utilisateur simple pour rejoindre le jeu. Lorsqu'un client tente de rejoindre une partie, il recherche automatiquement une session de jeu active avec un emplacement de joueur disponible. Si aucune session de jeu n'est disponible, le client demande une nouvelle session de jeu. Si une session de jeu est disponible, le client demande à rejoindre la session de jeu disponible. Lorsque vous testez votre jeu avec plusieurs clients simultanés, le premier client démarre la session de jeu et les autres clients rejoignent automatiquement la session de jeu existante.
- Sessions de jeu avec des machines à sous pour quatre joueurs. Vous pouvez lancer jusqu'à quatre instances de client de jeu simultanément et elles rejoindront la même session de jeu.

Lancer à partir d'un exécutable du serveur (facultatif)

Vous pouvez créer et lancer le fichier exécutable de votre serveur de jeu pour le tester sur une flotte Anywhere.

1. Créez un exécutable de serveur à l'aide du processus de construction standard de Unity. Dans Fichier, Paramètres de construction, passez de la plate-forme à un serveur dédié et créez.
2. Obtenez un jeton d'authentification à court terme en appelant la commande AWS CLI get-compute-auth-tokenavec votre identifiant de flotte Anywhere et votre AWS région. L'identifiant du parc est affiché dans Connect to an Anywhere Fleet lorsque vous créez le parc. La AWS région s'affiche dans Set Your Profile lorsque vous sélectionnez votre profil actif.

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

3. Lancez le nouveau fichier exécutable du serveur de jeu depuis une ligne de commande et transmettez un jeton d'authentification valide.

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

Déployez votre jeu sur un hébergement cloud avec des flottes EC2 gérées

Dans ce flux de travail, vous utilisez le plugin pour préparer votre jeu à l'hébergement sur des ressources informatiques basées sur le cloud gérées par Amazon GameLift. Vous ajoutez le code de jeu client et serveur pour les GameLift fonctionnalités Amazon, puis vous téléchargez la version de votre serveur sur le GameLift service Amazon pour l'hébergement. Lorsque ce flux de travail sera terminé, vous disposerez de serveurs de jeu exécutés dans le cloud et d'un client de jeu fonctionnel qui pourra s'y connecter.

Pour démarrer le flux de travail Amazon EC2 GameLift géré par Amazon :

- Dans le menu principal de l'éditeur Unity, choisissez Amazon, GameLift puis Host with Managed EC2. Ce flux de travail présente un processus en six étapes pour intégrer, créer, déployer et lancer les composants de votre jeu.

Définissez votre profil

Choisissez le profil que vous souhaitez utiliser lorsque vous suivez ce flux de travail. Le profil que vous sélectionnez a un impact sur toutes les étapes du flux de travail. Toutes les ressources que vous créez sont associées au AWS compte du profil et sont placées dans la AWS région par défaut du profil. Les autorisations de l'utilisateur du profil déterminent votre accès aux AWS ressources et aux actions.

1. Sélectionnez un profil dans la liste déroulante des profils disponibles. Si vous n'avez pas encore de profil ou si vous souhaitez en créer un nouveau, rendez-vous dans le GameLift menu Amazon et choisissez Set AWS Account Profiles.
2. Si le statut du bootstrap n'est pas « Actif », choisissez le profil Bootstrap et attendez que le statut passe à « Actif ».

Intégrez votre jeu à Amazon GameLift

Pour cette tâche, vous devez mettre à jour le code du client et du serveur dans votre projet de jeu.

- Les serveurs de jeu doivent être en mesure de communiquer avec le GameLift service Amazon pour recevoir des instructions pour démarrer une session de jeu, fournir des informations de connexion à une session de jeu et signaler le statut.
- Les clients du jeu doivent être en mesure d'obtenir des informations sur les sessions de jeu, de rejoindre ou de démarrer des sessions de jeu, et d'obtenir des informations de connexion pour rejoindre une partie.

> **Note**
>
> Si vous avez importé l'exemple de jeu, vous pouvez ignorer cette étape. Les exemples de ressources de jeu contiennent déjà le code serveur et client nécessaire.

Intégrez le code de votre serveur

Lorsque vous utilisez votre propre projet de jeu avec des scènes personnalisées, utilisez l'exemple de code fourni pour ajouter le code de serveur requis à votre projet de jeu. Si vous avez intégré votre projet de jeu à des fins de test à une flotte Anywhere, vous avez déjà suivi les instructions de cette étape.

1. Dans les fichiers de votre projet de jeu, ouvrez le `Assets/Scripts/Server` dossier. S'il n'existe pas, créez-le.
2. Accédez au GitHub repo aws/ amazon-gamelift-plugin-unity et ouvrez le chemin. `Samples~/SampleGame/Assets/Scripts/Server`
3. Localisez le fichier `GameLiftServer.cs` et copiez-le dans le `Server` dossier de votre projet de jeu. Lorsque vous créez un fichier exécutable pour le serveur, utilisez ce fichier comme cible de génération.

L'exemple de code inclut les éléments minimaux requis suivants, qui utilisent le SDK du serveur Amazon GameLift C# (version 5) :

- Initialise un client d' GameLift API Amazon. L'appel initSDK () avec les paramètres du serveur est requis pour une flotte Amazon Anywhere. GameLift Ces paramètres sont automatiquement définis pour être utilisés dans le plugin.
- Implémente les fonctions de rappel requises pour répondre aux demandes du GameLift service Amazon, notamment `OnStartGameSession``OnProcessTerminate`, et`onHealthCheck`.
- Appels `ProcessReady()` avec un port désigné pour informer le GameLift service Amazon lorsque le processus du serveur est prêt à héberger des sessions de jeu.

Si vous souhaitez personnaliser l'exemple de code du serveur, consultez les ressources suivantes :

- Ajoutez Amazon GameLift à votre serveur de jeu
- Référence GameLift du SDK Amazon Server 5.x pour C# et Unity

Intégrez votre code client

Pour les clients de jeux qui se connectent à des serveurs de jeu basés sur le cloud, il est recommandé d'utiliser un service principal côté client pour passer des appels vers le GameLift service Amazon, au lieu de passer les appels directement depuis le client du jeu.

Dans le flux de travail du plugin pour l'hébergement sur un parc EC2 géré, chaque scénario de déploiement inclut un service principal prédéfini qui inclut les composants suivants :

- Ensemble de fonctions Lambda et de tables DynamoDB utilisées pour demander des sessions de jeu et récupérer les informations relatives aux sessions de jeu. Ces composants utilisent une passerelle d'API comme proxy.

- Un groupe d'utilisateurs Amazon Cognito qui génère des identifiants de joueur uniques et authentifie les connexions des joueurs.

Pour utiliser ces composants, votre client de jeu a besoin de fonctionnalités lui permettant d'envoyer des demandes au service principal pour effectuer les opérations suivantes :

- Créez un utilisateur joueur dans le AWS groupe d'utilisateurs de Cognito et authentifiez-le.
- Participez à une session de jeu et recevez des informations de connexion.
- Rejoignez une partie en utilisant le matchmaking.

Utilisez les ressources suivantes comme guide.

- Intégrez le client à la GameLiftCoreApiclasse dans le GitHub dépôt amazon-gamelift-plugin-unityaws/. Cette classe fournit des contrôles pour l'authentification des joueurs et pour la récupération des informations de session de jeu.
- Pour voir les exemples d'intégrations de jeux, rendez-vous dans le GitHub dépôt amazon-gamelift-plugin-unityaws/,. `Samples~/SampleGame/Assets/Scripts/Client/GameLiftClient.cs`
- Ajoutez Amazon GameLift à votre client de jeu Unity.

Sélectionnez le scénario de déploiement

Au cours de cette étape, vous choisissez la solution d'hébergement de jeux que vous souhaitez déployer à ce stade. Vous pouvez effectuer plusieurs déploiements de votre jeu, en utilisant n'importe quel scénario.

- Flotte à région unique : déploie votre serveur de jeu sur une flotte unique de ressources d'hébergement dans la région par défaut AWS du profil actif. Ce scénario constitue un bon point de départ pour tester l'intégration de votre serveur AWS et la configuration de la version du serveur. Il déploie les ressources suivantes :
  - AWSflotte (à la demande) avec la version de votre serveur de jeu installée et en cours d'exécution.
  - Groupe d'utilisateurs et client Amazon Cognito pour permettre aux joueurs de s'authentifier et de démarrer une partie.
  - Autorisateur de passerelle d'API qui relie le groupe d'utilisateurs aux API.
  - WebACL pour limiter les appels excessifs des joueurs à la passerelle API.

- Passerelle API + fonction Lambda permettant aux joueurs de demander une machine à sous. Cette fonction appelle `CreateGameSession()` si aucune fonction n'est disponible.
- Passerelle API + fonction Lambda permettant aux joueurs d'obtenir des informations de connexion pour leur demande de jeu.

- FlexMatch flotte : déploie votre serveur de jeu sur un ensemble de flottes et met en place un FlexMatch système de matchmaking avec des règles pour créer des parties entre joueurs. Ce scénario utilise un hébergement Spot à faible coût avec une structure multi-flottes et multi-sites pour une disponibilité durable. Cette approche est utile lorsque vous êtes prêt à commencer à concevoir un composant de matchmaking pour votre solution d'hébergement. Dans ce scénario, vous allez créer les ressources de base pour cette solution, que vous pourrez personnaliser ultérieurement selon vos besoins. Il déploie les ressources suivantes :
  - FlexMatch configuration du matchmaking et règles de matchmaking définies pour accepter les demandes des joueurs et former des matchs.
  - Trois AWS flottes équipées de votre version de serveur de jeu sont installées et fonctionnent à plusieurs endroits. Comprend deux flottes Spot et une flotte à la demande en tant que sauvegarde.
  - AWSfile d'attente de placement de sessions de jeu qui répond aux demandes de matchs proposés en trouvant la meilleure ressource d'hébergement possible (en fonction de la viabilité, du coût, de la latence des joueurs, etc.) et en démarrant une session de jeu.
  - Groupe d'utilisateurs et client Amazon Cognito pour permettre aux joueurs de s'authentifier et de démarrer une partie.
  - Autorisateur de passerelle d'API qui relie le groupe d'utilisateurs aux API.
  - WebACL pour limiter les appels excessifs des joueurs à la passerelle API.
  - Passerelle API + fonction Lambda permettant aux joueurs de demander une machine à sous. Cette fonction appelle`StartMatchmaking()`.
  - Passerelle API + fonction Lambda permettant aux joueurs d'obtenir des informations de connexion pour leur demande de jeu.
  - Tables Amazon DynamoDB pour stocker les tickets de matchmaking pour les joueurs et les informations sur les sessions de jeu.
  - Rubrique SNS + Fonction Lambda pour GameSessionQueue gérer les événements.

Définissez les paramètres du jeu

Dans cette étape, vous décrivez le jeu vers lequel vous souhaitez le téléchargerAWS.

- Nom du jeu : Donnez un nom significatif à votre projet de jeu. Ce nom est utilisé dans le plugin.
- Nom du parc : donnez un nom significatif à votre parc EC2 géré. Amazon GameLift utilise ce nom (ainsi que l'ID de flotte) pour répertorier les ressources dans la AWS console.
- Nom de la version : donnez un nom significatif à la version de votre serveur. AWSutilise ce nom pour faire référence à la copie de la version de votre serveur qui est téléchargée sur Amazon GameLift et utilisée pour les déploiements.
- Paramètres de lancement : entrez les instructions facultatives à exécuter lors du lancement de l'exécutable du serveur sur une instance de flotte EC2 gérée. La longueur maximale est de 1024 caractères.
- Dossier du serveur de jeu : indiquez le chemin d'accès à un dossier local contenant la version de votre serveur.
- Fichier du serveur de jeu : Spécifiez le nom du fichier exécutable du serveur.

Scénario de déploiement

Au cours de cette étape, vous déployez votre jeu sur une solution d'hébergement cloud en fonction du scénario de déploiement que vous avez choisi. Ce processus peut prendre jusqu'à 40 minutes pour AWS valider la version de votre serveur, approvisionner les ressources d'hébergement, installer votre serveur de jeu, lancer les processus du serveur et les préparer à héberger des sessions de jeu.

Pour démarrer le déploiement, choisissez Deploy CloudFormation. Vous pouvez suivre l'état de votre hébergement de jeux ici. Pour obtenir des informations plus détaillées, vous pouvez vous connecter à la console AWS de gestion AWS et consulter les notifications d'événements. Assurez-vous de vous connecter en utilisant le même compte, le même utilisateur et AWS la même région que le profil utilisateur actif dans le plugin.

Lorsque le déploiement est terminé, votre serveur de jeu est installé sur une instance AWS EC2. Au moins un processus serveur est en cours d'exécution et prêt à démarrer une session de jeu.

Lancer le client du jeu

Lorsque votre flotte est déployée avec succès, des serveurs de jeu sont désormais opérationnels et disponibles pour héberger des sessions de jeu. Vous pouvez maintenant créer votre client, le lancer, vous connecter pour rejoindre la session de jeu.

1. Configurez votre client de jeu. Au cours de cette étape, vous demandez au plugin de mettre à jour une `GameLiftClientSettings` ressource pour votre projet de jeu. Le plugin utilise cet

actif pour stocker certaines informations dont votre client de jeu a besoin pour se connecter au GameLift service Amazon.

   a. Si vous n'avez pas importé et initialisé le jeu d'exemple, créez une nouvelle `GameLiftClientSettings` ressource. Dans le menu principal de l'éditeur Unity, choisissez Assets, Create GameLift, Client Settings. Si vous créez plusieurs copies de GameLiftClientSettings votre projet, le plugin le détecte automatiquement et vous indique quel actif le plugin va mettre à jour.

   b. Dans Launch Game, choisissez Configurer le client : appliquer les paramètres EC2 gérés. Cette action met à jour les paramètres de votre client de jeu afin d'utiliser le parc EC2 géré que vous venez de déployer.

2. Créez votre client de jeu. Créez un exécutable client en utilisant le processus de construction standard de Unity. Dans Fichier, Paramètres de compilation, basculez la plateforme sur Windows, Mac, Linux. Si vous avez importé l'exemple de jeu et initialisé les paramètres, la liste des builds et la cible des builds sont automatiquement mis à jour.
3. Lancez le nouveau fichier exécutable du client de jeu. Pour commencer à jouer, lancez deux à quatre instances clientes et utilisez l'interface utilisateur de chacune pour rejoindre une session de jeu.

Si vous utilisez l'exemple de client de jeu, il présente les caractéristiques suivantes :

- Un composant de connexion des joueurs. Lorsque vous vous connectez à un serveur de jeu d'une flotte Anywhere, aucune validation n'est requise par le joueur. Vous pouvez saisir n'importe quelle valeur pour rejoindre la session de jeu.
- Une interface utilisateur simple pour rejoindre le jeu. Lorsqu'un client tente de rejoindre une partie, il recherche automatiquement une session de jeu active avec un emplacement de joueur disponible. Si aucune session de jeu n'est disponible, le client demande une nouvelle session de jeu. Si une session de jeu est disponible, le client demande à rejoindre la session de jeu disponible. Lorsque vous testez votre jeu avec plusieurs clients simultanés, le premier client démarre la session de jeu et les autres clients rejoignent automatiquement la session de jeu existante.
- Sessions de jeu avec des machines à sous pour quatre joueurs. Vous pouvez lancer jusqu'à quatre instances de client de jeu simultanément et elles rejoindront la même session de jeu.

Guide du GameLift plugin Amazon pour Unity pour le SDK de serveur 4.x

> **Note**
>
> Cette rubrique fournit des informations relatives à une version antérieure du GameLift plugin Amazon pour Unity. La version 1.0.0 (publiée en 2021) utilise le SDK du GameLift serveur Amazon 4.x ou une version antérieure. Pour obtenir de la documentation sur la dernière version du plugin, qui utilise le SDK 5.x du serveur et prend en charge Amazon GameLift Anywhere, consultez. Guide du GameLift plugin Amazon pour Unity pour le SDK de serveur 5.x

Amazon GameLift fournit des outils pour préparer vos serveurs de jeux multijoueurs à fonctionner sur Amazon GameLift. Le GameLift plugin Amazon pour Unity facilite l'intégration d'Amazon GameLift dans vos projets de jeux Unity et le déploiement des GameLift ressources Amazon pour l'hébergement dans le cloud. Utilisez le plugin Unity pour accéder aux GameLift API Amazon et déployer des AWS CloudFormation modèles pour des scénarios de jeu courants.

Après avoir configuré le plugin, vous pouvez essayer l'exemple Amazon GameLift Unity sur GitHub.

Rubriques

Intégrer Amazon GameLift à un projet de serveur de jeu Unity

> **Note**
>
> Cette rubrique fournit des informations relatives à une version antérieure du GameLift plugin Amazon pour Unity. La version 1.0.0 (publiée en 2021) utilise le SDK du GameLift serveur

Amazon 4.x ou une version antérieure. Pour obtenir de la documentation sur la dernière version du plugin, qui utilise le SDK 5.x du serveur et prend en charge Amazon GameLift Anywhere, consultez. Guide du GameLift plugin Amazon pour Unity pour le SDK de serveur 5.x

Cette rubrique vous aide à préparer votre serveur de jeu personnalisé pour l'hébergement sur Amazon GameLift. Le serveur de jeu doit être en mesure d' GameLift informer Amazon de son statut, de démarrer et d'arrêter des sessions de jeu lorsqu'il y est invité, et d'effectuer d'autres tâches. Pour plus d'informations, consultez Ajoutez Amazon GameLift à votre serveur de jeu.

Prérequis

Avant d'intégrer votre serveur de jeu, effectuez les tâches suivantes :

- Configurer un rôle de service IAM pour Amazon GameLift
- Installez le plugin pour Unity

Configuration d'un nouveau processus serveur

> **Note**
>
> Cette rubrique fait référence au GameLift plugin Amazon pour Unity version 1.0.0, qui utilise le SDK de serveur 4.x ou une version antérieure.

Configurez la communication avec Amazon GameLift et signalez que le processus du serveur est prêt à héberger une session de jeu.

1. Initialisez le SDK du serveur en appelant. `InitSDK()`
2. Pour préparer le serveur à accepter une session de jeu, appelez `ProcessReady()` avec le port de connexion et les détails de l'emplacement de la session de jeu. Incluez les noms des fonctions de rappel invoquées par le GameLift service Amazon, telles que`OnGameSession()`,,`OnGameSessionUpdate()`,`OnProcessTerminate()`. `OnHealthCheck()` Amazon GameLift peut prendre quelques minutes pour vous rappeler.
3. Amazon GameLift met à jour l'état du processus du serveur à`ACTIVE`.
4. Amazon GameLift appelle `onHealthCheck` régulièrement.

L'exemple de code suivant montre comment configurer un processus serveur simple avec 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;
}
```

Démarrer une session de jeu

> **Note**
>
> Cette rubrique fait référence au GameLift plugin Amazon pour Unity version 1.0.0, qui utilise le SDK de serveur 4.x ou une version antérieure.

Une fois l'initialisation du jeu terminée, vous pouvez démarrer une session de jeu.

1. Implémentez la fonction de rappel `onStartGameSession`. Amazon GameLift utilise cette méthode pour démarrer une nouvelle session de jeu sur le serveur et recevoir les connexions des joueurs.
2. Pour activer une session de jeu, appelez`ActivateGameSession()`. Pour plus d'informations sur le SDK, consultezRéférence GameLift du SDK Amazon Server (C#) : Actions.

L'exemple de code suivant montre comment démarrer une session de jeu avec 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();
}
```

Fin d'une session de jeu

> **Note**
>
> Cette rubrique fait référence au GameLift plugin Amazon pour Unity version 1.0.0, qui utilise le SDK de serveur 4.x ou une version antérieure.

Avertissez Amazon GameLift lorsqu'une session de jeu se termine. Il est recommandé d'arrêter les processus du serveur une fois les sessions de jeu terminées afin de recycler et d'actualiser les ressources d'hébergement.

1. Configurez une fonction nommée on`ProcessTerminate` pour recevoir les demandes d'Amazon GameLift et appelez`ProcessEnding()`.
2. L'état du processus passe à`TERMINATED`.

L'exemple suivant décrit comment mettre fin à un processus pour une session de jeu.

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

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

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

Création d'un serveur, création et téléchargement sur Amazon GameLift

> **Note**
>
> Cette rubrique fait référence au GameLift plugin Amazon pour Unity version 1.0.0, qui utilise le SDK de serveur 4.x ou une version antérieure.

Après avoir intégré votre serveur de jeu à Amazon GameLift, téléchargez les fichiers de compilation dans une flotte afin qu'Amazon GameLift puisse les déployer pour l'hébergement de jeux. Pour plus d'informations sur le chargement de votre serveur sur Amazon GameLift, consultezTéléchargez une version de serveur personnalisée sur Amazon GameLift.

Intégrer Amazon GameLift à un projet client de jeu Unity

> **Note**
>
> Cette rubrique fournit des informations relatives à une version antérieure du GameLift plugin Amazon pour Unity. La version 1.0.0 (publiée en 2021) utilise le SDK du GameLift serveur Amazon 4.x ou une version antérieure. Pour obtenir de la documentation sur la dernière

version du plugin, qui utilise le SDK 5.x du serveur et prend en charge Amazon GameLift Anywhere, consultez. Guide du GameLift plugin Amazon pour Unity pour le SDK de serveur 5.x

Cette rubrique vous aide à configurer un client de jeu pour qu'il se connecte aux sessions de jeu GameLift hébergées par Amazon via un service principal. Utilisez GameLift les API Amazon pour lancer le matchmaking, demander le placement d'une session de jeu, etc.

Ajoutez du code au projet de service principal pour permettre la communication avec le GameLift service Amazon. Un service principal gère toutes les communications du client du jeu avec le GameLift service. Pour plus d'informations sur les services principaux, consultezConcevez le service client de votre jeu.

Un serveur principal gère les tâches suivantes du client de jeu :

- Personnalisez l'authentification pour vos joueurs.
- Demandez des informations sur les sessions de jeu actives auprès du GameLift service Amazon.
- Créez une nouvelle session de jeu.
- Ajoutez un joueur à une session de jeu existante.
- Supprimer un joueur d'une session de jeu existante.

Rubriques

Prérequis

Avant de configurer la communication entre le serveur de jeu et le GameLift client Amazon, effectuez les tâches suivantes :

- Configurez un Compte AWS

- Installez le plugin pour Unity
- Intégrer Amazon GameLift à un projet de serveur de jeu Unity
- Configuration des GameLift flottes Amazon

Initialisation d'un client de jeu

> **Note**
>
> Cette rubrique fait référence au GameLift plugin Amazon pour Unity version 1.0.0, qui utilise le SDK de serveur 4.x ou une version antérieure.

Ajoutez du code pour initialiser un client de jeu. Exécutez ce code au lancement, il est nécessaire pour les autres GameLift fonctions d'Amazon.

1. Initialiser`AmazonGameLiftClient`. Appelez `AmazonGameLiftClient` avec une configuration client par défaut ou une configuration personnalisée. Pour plus d'informations sur la configuration d'un client, consultezConfigurer Amazon GameLift sur un service de backend.
2. Générez un identifiant de joueur unique pour que chaque joueur se connecte à une session de jeu. Pour plus d'informations, consultez Générer des identifiants de joueurs.

   Les exemples suivants montrent comment configurer un GameLift client 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);

    }
}
```

Créer une session de jeu sur une flotte spécifique

> **Note**
>
> Cette rubrique fait référence au GameLift plugin Amazon pour Unity version 1.0.0, qui utilise le SDK de serveur 4.x ou une version antérieure.

Ajoutez du code pour démarrer de nouvelles sessions de jeu sur vos flottes déployées et les mettre à disposition des joueurs. Une fois GameLift qu'Amazon a créé la nouvelle session de jeu et renvoyé une`GameSession`, vous pouvez y ajouter des joueurs.

- Faites une demande pour une nouvelle session de jeu.
  - Si votre jeu utilise des flottes, appelez `CreateGameSession()` avec un identifiant de flotte ou un alias, un nom de session et le nombre maximum de joueurs simultanés pour le jeu.
  - Si votre jeu utilise des files d'attente, appelez`StartGameSessionPlacement()`.

L'exemple suivant montre comment créer une session de jeu.

```
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;
}
```

Ajouter des joueurs aux sessions de jeu

> **Note**
>
> Cette rubrique fait référence au GameLift plugin Amazon pour Unity version 1.0.0, qui utilise le SDK de serveur 4.x ou une version antérieure.

Une fois GameLift qu'Amazon a créé la nouvelle session de jeu et renvoyé un `GameSession` objet, vous pouvez y ajouter des joueurs.

1. Réservez une place de joueur dans une session de jeu en créant une nouvelle session de joueur. Utilisez `CreatePlayerSession` ou `CreatePlayerSessions` avec l'identifiant de session de jeu et un identifiant unique pour chaque joueur.
2. Connectez-vous à la session de jeu. Récupérez l'`PlayerSession`objet pour obtenir les informations de connexion de la session de jeu. Vous pouvez utiliser ces informations pour établir une connexion directe avec le processus du serveur :
    a. Utilisez le port spécifié et le nom DNS ou l'adresse IP du processus serveur.
    b. Utilisez le nom DNS et le port de vos flottes. Le nom et le port DNS sont obligatoires si la génération de certificats TLS est activée dans vos flottes.
    c. Référencez l'ID de session du joueur. L'identifiant de session du joueur est requis si votre serveur de jeu valide les connexions entrantes des joueurs.

Les exemples suivants montrent comment réserver une place de joueur dans une session de jeu.

```
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;
}
```

Le code suivant montre comment connecter un joueur à la session de jeu.

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

Supprimer un joueur d'une session de jeu

> **Note**
>
> Cette rubrique fait référence au GameLift plugin Amazon pour Unity version 1.0.0, qui utilise le SDK de serveur 4.x ou une version antérieure.

Vous pouvez retirer les joueurs de la session de jeu lorsqu'ils quittent le jeu.

1. Informez le GameLift service Amazon qu'un joueur s'est déconnecté du processus du serveur. Appelez `RemovePlayerSession` avec l'identifiant de session du joueur.
2. Vérifiez que cela `RemovePlayerSession` revient`Success`. Amazon GameLift modifie ensuite l'emplacement de joueur qui sera disponible, qu'Amazon GameLift peut attribuer à un nouveau joueur.

L'exemple suivant montre comment supprimer une session de joueur.

```
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());
    }
}
```

Installation et configuration du plugin

> **Note**
>
> Cette rubrique fait référence au GameLift plugin Amazon pour Unity version 1.0.0, qui utilise le SDK de serveur 4.x ou une version antérieure.

Cette section explique comment télécharger, installer et configurer le GameLift plugin Amazon pour Unity, version 1.0.0.

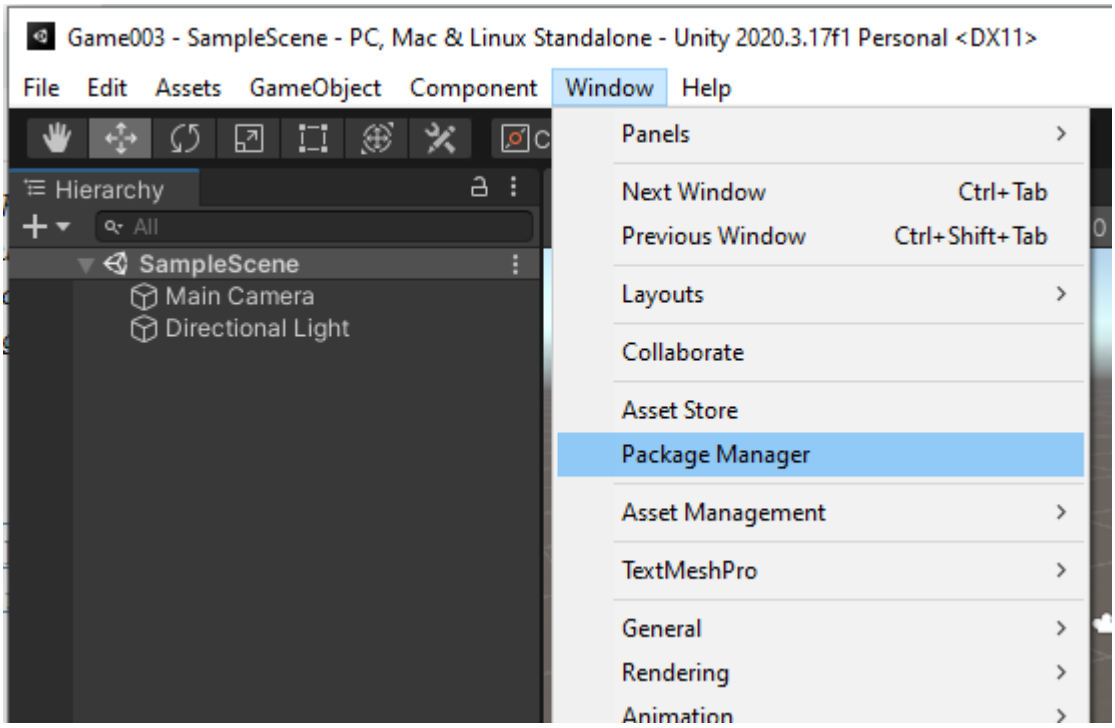
Prérequis

- Unity pour Windows 2019.4 LTS, Windows 2020.3 LTS ou Unity pour macOS
- Version actuelle de Java
- Version actuelle de .NET 4.x

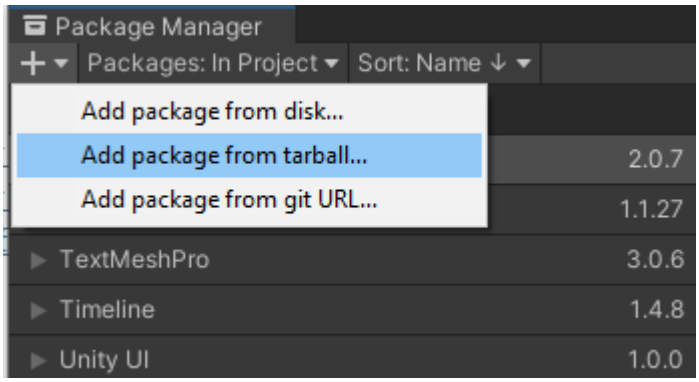
Pour télécharger et installer le plugin pour Unity

1. Téléchargez le GameLift plugin Amazon pour Unity. Vous trouverez la dernière version sur la page du référentiel du GameLift plugin Amazon pour Unity. Dans la dernière version, choisissez Assets, puis téléchargez le `com.amazonaws.gamelift-`*`version`*`.tgz` fichier.
2. Lancez Unity et choisissez un projet.

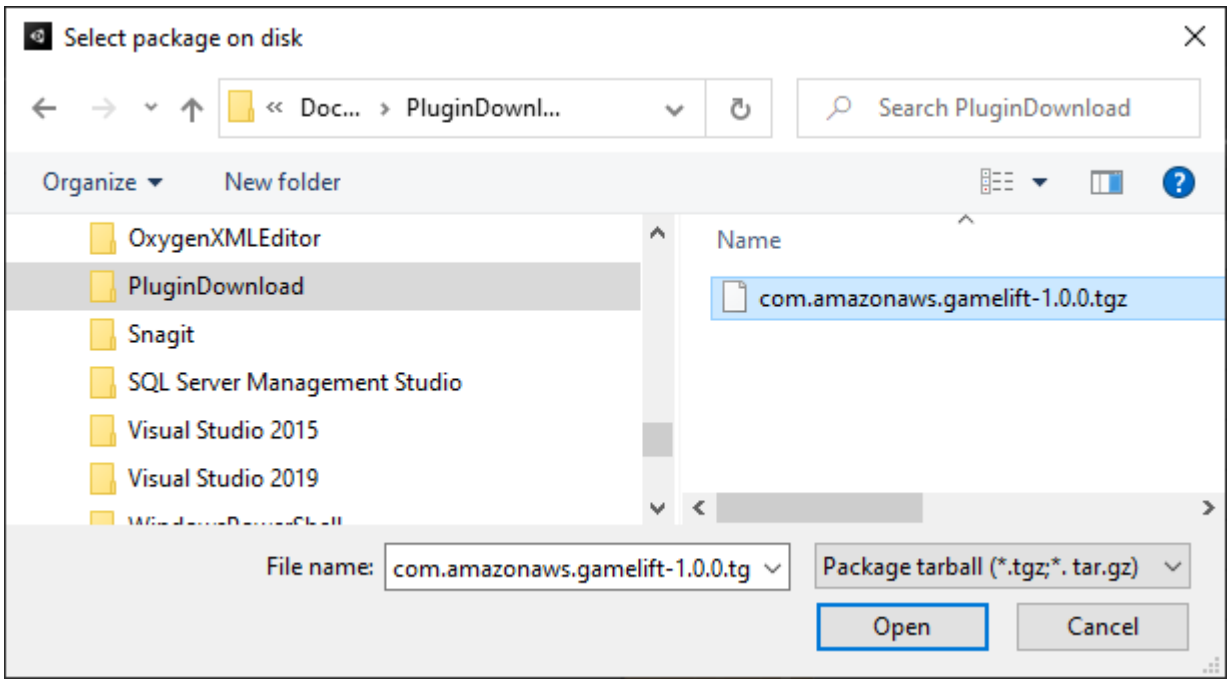
3. Dans la barre de navigation supérieure, sous Window, choisissez Package Manager :



4. Dans l'onglet Gestionnaire de packages, choisissez +, puis choisissez Ajouter un package depuis une archive... :



5. Dans la fenêtre Sélectionner les packages sur le disque, accédez au `com.amazonaws.gamelift` dossier, choisissez le fichier`com.amazonaws.gamelift-`*`version`*`.tgz` , puis choisissez Ouvrir :



6. Une fois que Unity a chargé le plug-in, Amazon GameLift apparaît en tant que nouvel élément dans le menu Unity. L'installation et la recompilation des scripts peuvent prendre quelques minutes. L'onglet Amazon GameLift Plugin Settings s'ouvre automatiquement.

7. Dans le volet SDK, choisissez Utiliser .NET 4.x.

   Une fois configuré, le statut passe de Non configuré à Configuré.

Testez votre jeu localement

> **Note**
>
> Cette rubrique fait référence au GameLift plugin Amazon pour Unity version 1.0.0, qui utilise le SDK de serveur 4.x ou une version antérieure.

Utilisez Amazon GameLift Local pour exécuter Amazon GameLift sur votre appareil local. Vous pouvez utiliser Amazon GameLift Local pour vérifier les modifications de code en quelques secondes, sans connexion réseau.

Configuration des tests locaux

1. Dans la fenêtre du plugin pour Unity, choisissez l'onglet Test.
2. Dans le volet Test, choisissez Download Amazon GameLift Local. Le plugin pour Unity ouvre une fenêtre de navigateur et télécharge le `GameLift_06_03_2021.zip` fichier dans votre dossier de téléchargements.

   Le téléchargement inclut le SDK du serveur C#, les fichiers source .NET et un composant .NET compatible avec Unity.
3. Décompressez le fichier `GameLift_06_03_2021.zip` téléchargé.
4. Dans la fenêtre Amazon GameLift Plugin Settings, choisissez Amazon GameLift Local Path, accédez au dossier décompressé, choisissez le fichier**`GameLiftLocal.jar`**, puis choisissez Ouvrir.

   Une fois configuré, le statut des tests locaux passe de Non configuré à Configuré.
5. Vérifiez l'état du JRE. Si le statut est Non configuré, choisissez Télécharger JRE et installez la version Java recommandée.

   Après avoir installé et configuré l'environnement Java, le statut passe à Configuré.

Lancez votre jeu local

1. Dans l'onglet Plugin pour Unity, choisissez l'onglet Test.
2. Dans le volet Test, choisissez Open Local Test UI.

3. Dans la fenêtre de test local, spécifiez le chemin exécutable du serveur. Sélectionnez... pour sélectionner le chemin et le nom du fichier exécutable de votre application serveur.
4. Dans la fenêtre de test local, spécifiez un port GL local.
5. Choisissez Deploy & Run pour déployer et exécuter le serveur.
6. Pour arrêter votre serveur de jeu, choisissez Arrêter ou fermez les fenêtres du serveur de jeu.

Déployer un scénario

> **Note**
>
> Cette rubrique fait référence au GameLift plugin Amazon pour Unity version 1.0.0, qui utilise le SDK de serveur 4.x ou une version antérieure.

Un scénario utilise un AWS CloudFormation modèle pour créer les ressources dont vous avez besoin pour déployer une solution d'hébergement cloud pour votre jeu. Cette section décrit les scénarios proposés par GameLift Amazon et explique comment les utiliser.

Prérequis

Pour déployer le scénario, vous avez besoin d'un rôle IAM pour le GameLift service Amazon. Pour plus d'informations sur la création d'un rôle pour Amazon GameLift, consultezConfigurez un Compte AWS.

Chaque scénario nécessite des autorisations d'accès aux ressources suivantes :

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

Scénarios

> **Note**
>
> Cette rubrique fait référence au GameLift plugin Amazon pour Unity version 1.0.0, qui utilise le SDK de serveur 4.x ou une version antérieure.

Le GameLift plug-in Amazon pour Unity inclut les scénarios suivants :

Authentification uniquement

Ce scénario crée un service de backend de jeu qui effectue l'authentification des joueurs sans capacité de serveur de jeu. Le modèle crée les ressources suivantes dans votre compte :

- Un groupe d'utilisateurs Amazon Cognito pour stocker les informations d'authentification des joueurs.
- Un AWS Lambda gestionnaire basé sur un point de terminaison Amazon API Gateway REST qui démarre des jeux et affiche les informations de connexion aux jeux.

Flotte d'une seule région

Ce scénario crée un service de backend de jeu avec une seule GameLift flotte Amazon. Il crée les ressources suivantes :

- Un pool d'utilisateurs Amazon Cognito permettant à un joueur de s'authentifier et de démarrer une partie.
- Un AWS Lambda gestionnaire chargé de rechercher une session de jeu existante avec un emplacement de joueur libre dans la flotte. S'il ne trouve pas d'emplacement libre, il crée une nouvelle session de jeu.

Une flotte multirégionale avec une file d'attente et un système de matchmaking personnalisé

Ce scénario forme des matchs en utilisant les GameLift files d'attente Amazon et un système de matchmaking personnalisé pour regrouper les joueurs les plus âgés du pool d'attente. Il crée les ressources suivantes :

- Rubrique Amazon Simple Notification Service sur laquelle Amazon GameLift publie des messages. Pour plus d'informations sur les sujets et les notifications liés aux réseaux sociaux, consultezConfigurer une notification d'événement pour le placement des sessions de jeu.
- Fonction Lambda invoquée par le message qui communique les détails du placement et de la connexion au jeu.
- Une table Amazon DynamoDB pour stocker les informations relatives à l'emplacement et à la connexion au jeu. `GetGameConnection`les appels sont lus dans ce tableau et renvoient les informations de connexion au client du jeu.

Repérez les flottes grâce à une file d'attente et à un système de matchmaking personnalisé

Ce scénario forme des correspondances en utilisant les GameLift files d'attente Amazon et un système de matchmaking personnalisé et configure trois flottes. Il crée les ressources suivantes :

- Deux flottes Spot contenant différents types d'instances afin de garantir la durabilité en cas d'indisponibilité de Spot.
- Une flotte à la demande qui fait office de sauvegarde pour les autres flottes Spot. Pour plus d'informations sur la conception de vos flottes, consultezGuide GameLift de conception de flotte Amazon.
- Une GameLift file d'attente Amazon pour maintenir la disponibilité des serveurs à un niveau élevé et à un faible coût. Pour plus d'informations et les meilleures pratiques concernant les files d'attente, consultezConception d'une file d'attente de sessions de jeu.

FlexMatch

Ce scénario utilise FlexMatch, un service de matchmaking géré, pour associer les joueurs. Pour plus d'informations FlexMatch, consultez What is Amazon GameLift FlexMatch. Ce scénario crée les ressources suivantes :

- Une fonction Lambda pour créer un ticket de matchmaking après réception `StartGame` de demandes.
- Une fonction Lambda séparée pour écouter les événements FlexMatch du match.

Pour éviter des frais inutilesCompte AWS, supprimez les ressources créées par chaque scénario une fois que vous avez fini de les utiliser. Supprimez la AWS CloudFormation pile correspondante.

Mettre à jour AWS les identifiants

> **Note**
>
> Cette rubrique fait référence au GameLift plugin Amazon pour Unity version 1.0.0, qui utilise le SDK de serveur 4.x ou une version antérieure.

Le GameLift plugin Amazon pour Unity nécessite des informations de sécurité pour déployer un scénario. Vous pouvez créer de nouvelles informations d'identification ou utiliser des informations d'identification existantes.

Pour plus d'informations sur la configuration des informations d'identification, voir Comprendre et obtenir vos AWS informations d'identification.

Pour mettre à jour AWS les identifiants

1. Dans Unity, dans l'onglet Plug-in pour Unity, choisissez l'onglet Déployer.
2. Dans le volet Déployer, sélectionnez AWSCredentials.
3. Vous pouvez créer de nouvelles AWS informations d'identification ou choisir des informations d'identification existantes.
   - Pour créer des informations d'identification, choisissez Créer un nouveau profil d'informations d'identification, puis spécifiez le nouveau nom du profil, AWSl'ID de clé d'accès, la clé AWS secrète et Région AWS.
   - Pour choisir des informations d'identification existantes, choisissez Choisir un profil d'informations d'identification existant, puis sélectionnez un nom de profil et Région AWS.
4. Dans la fenêtre Mettre à jour les AWS informations d'identification, choisissez Mettre à jour le profil des informations d'identification.

Mettre à jour le bootstrap du compte

> **Note**
>
> Cette rubrique fait référence au GameLift plugin Amazon pour Unity version 1.0.0, qui utilise le SDK de serveur 4.x ou une version antérieure.

L'emplacement du bootstrap est un compartiment Amazon S3 utilisé lors du déploiement. Il est utilisé pour stocker les actifs du serveur de jeu et d'autres dépendances. Le compartiment Région AWS que vous choisissez doit être la même région que celle que vous utiliserez pour le déploiement du scénario.

Pour plus d'informations sur les compartiments Amazon S3, consultez Création, configuration et utilisation des compartiments Amazon Simple Storage Service.

Pour mettre à jour l'emplacement du bootstrap du compte

1. Dans Unity, dans l'onglet Plug-in pour Unity, choisissez l'onglet Déployer.
2. Dans le volet Déploiement, choisissez Update Account Bootstrap.
3. Dans la fenêtre Account Bootstrapping, vous choisissez un compartiment Amazon S3 existant ou vous créez un nouveau compartiment Amazon S3 :
   - Pour choisir un compartiment existant, choisissez Choisir un compartiment Amazon S3 existant et Mettre à jour pour enregistrer votre sélection.
   - Choisissez Créer un nouveau compartiment Amazon S3 pour créer un nouveau compartiment Amazon Simple Storage Service, puis choisissez une politique. La politique précise la date d'expiration du compartiment Amazon S3. Choisissez Create pour créer le bucket.

Déployer un scénario de jeu

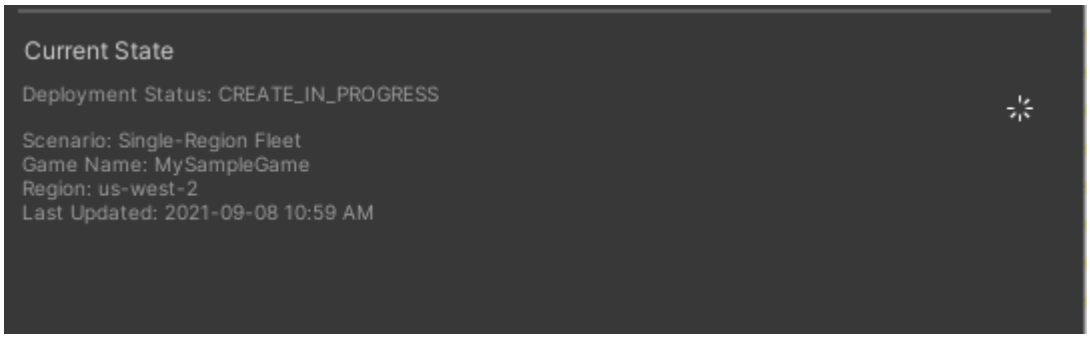
> **Note**
>
> Cette rubrique fait référence au GameLift plugin Amazon pour Unity version 1.0.0, qui utilise le SDK de serveur 4.x ou une version antérieure.

Vous pouvez utiliser un scénario pour tester votre jeu avec Amazon GameLift. Chaque scénario utilise un AWS CloudFormation modèle pour créer une pile avec les ressources requises. La plupart des scénarios nécessitent un exécutable sur le serveur de jeu et un chemin de compilation. Lorsque vous déployez le scénario, Amazon GameLift copie les ressources du jeu sur l'emplacement de démarrage dans le cadre du déploiement.

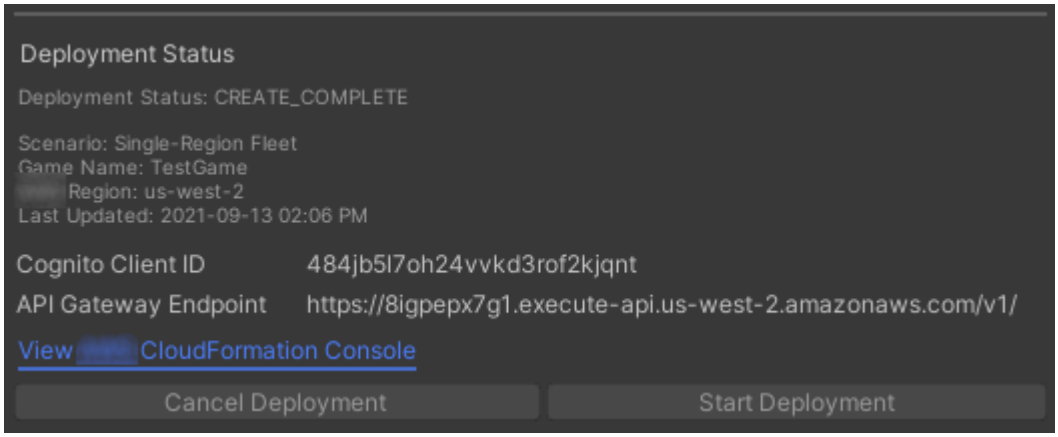
Vous devez configurer des AWS informations d'identification et un bootstrap de AWS compte pour déployer un scénario.

Pour déployer un scénario

1. Dans Unity, dans l'onglet Plug-in pour Unity, choisissez l'onglet Déployer.
2. Dans le volet Déploiement, choisissez Open Deployment UI.
3. Dans la fenêtre Déploiement, choisissez un scénario.
4. Entrez un nom de jeu. Il doit être unique. Le nom du jeu fait partie du nom de la AWS CloudFormation pile lorsque vous déployez le scénario.
5. Choisissez le chemin du dossier de compilation du serveur de jeu. Le chemin du dossier de construction pointe vers le dossier contenant le fichier exécutable du serveur et ses dépendances.
6. Choisissez le chemin du fichier .exe de compilation du serveur de jeu. Le chemin du fichier exécutable de compilation pointe vers le fichier exécutable du serveur de jeu.
7. Choisissez Démarrer le déploiement pour commencer à déployer un scénario. Vous pouvez suivre l'état de la mise à jour dans la fenêtre Déploiement sous État actuel. Le déploiement des scénarios peut prendre jusqu'à 30 minutes.



8. Lorsque le scénario est terminé, l'état actuel est mis à jour pour inclure l'ID du client Cognito et le point de terminaison API Gateway que vous pouvez copier et coller dans le jeu.



9. Pour mettre à jour les paramètres du jeu, dans le menu Unity, choisissez Accéder aux paramètres de connexion du client. Cela affiche un onglet Inspector sur le côté droit de l'écran Unity.
10. Désélectionnez le mode de test local.
11. Entrez le point de terminaison API Gateway et l'ID du client Coginito. Choisissez le même Région AWS que celui que vous avez utilisé pour le déploiement du scénario. Vous pouvez ensuite reconstruire et exécuter le client du jeu à l'aide des ressources du scénario déployées.

Supprimer les ressources créées par le scénario

> **Note**
>
> Cette rubrique fait référence au GameLift plugin Amazon pour Unity version 1.0.0, qui utilise le SDK de serveur 4.x ou une version antérieure.

Pour supprimer les ressources créées pour le scénario, supprimez la AWS CloudFormation pile correspondante.

Pour supprimer les ressources créées par le scénario

1. Dans la fenêtre de déploiement du GameLift plugin Amazon pour Unity, sélectionnez Afficher AWS CloudFormation la console pour ouvrir la AWS CloudFormation console.
2. Dans la AWS CloudFormation console, choisissez Stacks, puis choisissez la pile qui inclut le nom du jeu spécifié lors du déploiement.
3. Choisissez Supprimer pour supprimer la pile. La suppression d'une pile peut prendre quelques minutes. Après avoir AWS CloudFormation supprimé la pile utilisée par le scénario, son statut passe à`ROLLBACK_COMPLETE`.

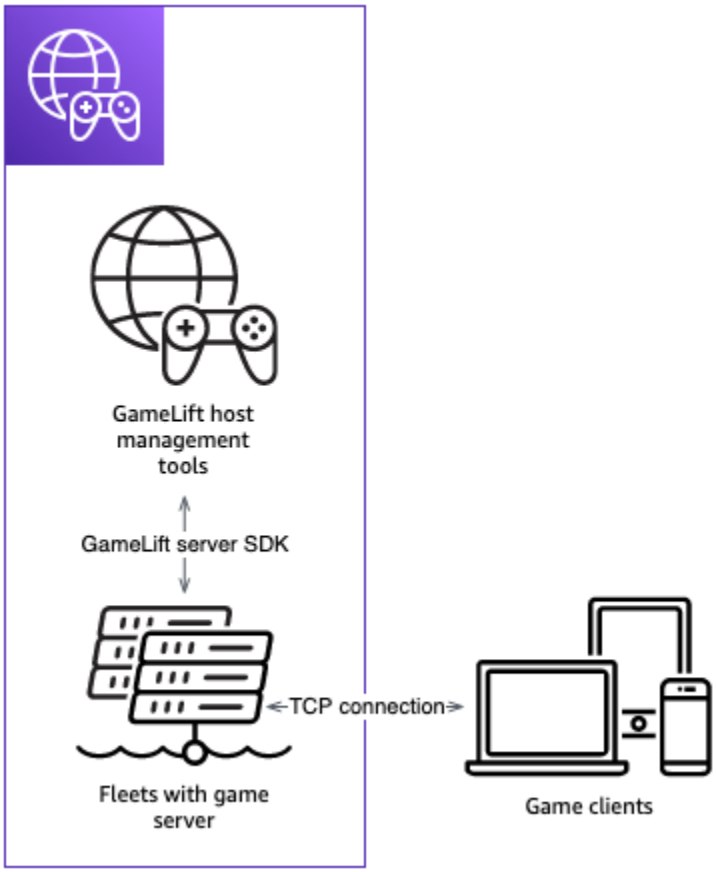
Intégrer des jeux à Amazon GameLift dans Unity

> **Note**
>
> Cette rubrique fait référence au GameLift plugin Amazon pour Unity version 1.0.0, qui utilise le SDK de serveur 4.x ou une version antérieure.

Intégrez votre jeu Unity à Amazon GameLift en effectuant les tâches suivantes :

- Intégrer Amazon GameLift à un projet de serveur de jeu Unity
- Intégrer Amazon GameLift à un projet client de jeu Unity

Le schéma suivant montre un exemple de flux d'intégration d'un jeu. Dans le schéma, une flotte avec le serveur de jeu est déployée sur Amazon GameLift. Le client du jeu communique avec le serveur de jeu, qui communique avec Amazon GameLift.



Importez et lancez un exemple de jeu

> **Note**
>
> Cette rubrique fait référence au GameLift plugin Amazon pour Unity version 1.0.0, qui utilise le SDK de serveur 4.x ou une version antérieure.

Le GameLift plugin Amazon pour Unity inclut un exemple de jeu que vous pouvez utiliser pour découvrir les bases de l'intégration de votre jeu à Amazon GameLift. Dans cette section, vous allez créer le client de jeu et le serveur de jeu, puis les tester localement à l'aide d'Amazon GameLift Local.

Prérequis

- Configurez un Compte AWS
- Installation et configuration du plugin

Créez et exécutez l'exemple de serveur de jeu

> **Note**
>
> Cette rubrique fait référence au GameLift plugin Amazon pour Unity version 1.0.0, qui utilise le SDK de serveur 4.x ou une version antérieure.

Configurez les fichiers du serveur de jeu de l'exemple de jeu.

1. Dans Unity, dans le menu, choisissez Amazon GameLift, puis choisissez Importer un échantillon de jeu.
2. Dans la fenêtre Importer un exemple de jeu, choisissez Importer pour importer le jeu, ses actifs et ses dépendances.
3. Construisez le serveur de jeu. Dans Unity, dans le menu, choisissez Amazon GameLift, puis choisissez Appliquer les paramètres de build de Windows Sample Server ou Appliquer les paramètres de build de macOS Sample Server. Après avoir configuré les paramètres du serveur de jeu, Unity recompile les ressources.
4. Dans Unity, dans le menu, choisissez Fichier, puis Build. Choisissez Server Build, choisissez Build, puis choisissez un dossier de build spécifiquement pour les fichiers du serveur.

   Unity construit l'exemple de serveur de jeu en plaçant le fichier exécutable et les ressources requises dans le dossier de compilation spécifié.

Créez et exécutez l'exemple de client de jeu

> **Note**
>
> Cette rubrique fait référence au GameLift plugin Amazon pour Unity version 1.0.0, qui utilise le SDK de serveur 4.x ou une version antérieure.

Configurez les fichiers client du jeu d'exemple.

1. Dans Unity, dans le menu, choisissez Amazon GameLift, puis choisissez Appliquer les paramètres de génération du client d'exemple de Windows ou Appliquer les paramètres de génération du client d'exemple de macOS. Une fois les paramètres du client de jeu configurés, Unity recompile les actifs.
2. Dans Unity, dans le menu, sélectionnez Accéder aux paramètres du client. Cela affichera un onglet Inspector sur le côté droit de l'écran Unity. Dans l'onglet Amazon GameLift Client Settings, sélectionnez le mode de test local.
3. Créez le client du jeu. Dans Unity, dans le menu, choisissez Fichier. Vérifiez que Server Build n'est pas cochée, choisissez Build, puis sélectionnez un dossier de build spécifiquement pour les fichiers client.

   Unity crée l'exemple de client de jeu en plaçant l'exécutable et les ressources requises dans le dossier de construction du client spécifié.
4. Vous n'avez pas créé le serveur et le client de jeu. Dans les étapes suivantes, vous lancerez le jeu et vous verrez comment il interagit avec Amazon GameLift.

Testez l'exemple de jeu localement

> **Note**
>
> Cette rubrique fait référence au GameLift plugin Amazon pour Unity version 1.0.0, qui utilise le SDK de serveur 4.x ou une version antérieure.

Exécutez l'exemple de jeu que vous avez importé à l'aide d'Amazon GameLift Local.

1. Lancez le serveur de jeu. Dans Unity, dans l'onglet Plug-in pour Unity, choisissez l'onglet Déployer.

2. Dans le volet Test, choisissez Open Local Test UI.
3. Dans la fenêtre de test local, spécifiez le chemin du fichier .exe du serveur de jeu. Le chemin doit inclure le nom de l'exécutable. Par exemple, `C:/MyGame/GameServer/MyGameServer.exe`.
4. Choisissez Deploy and Run. Le plugin pour Unity lance le serveur de jeu et ouvre une fenêtre de journal Amazon GameLift Local. La fenêtre contient des messages de journal, notamment des messages envoyés entre le serveur de jeu et Amazon GameLift Local.
5. Lancez le client du jeu. Trouvez l'emplacement de compilation à l'aide de l'exemple de client de jeu et choisissez le fichier exécutable.
6. Dans le jeu Amazon GameLift Sample, saisissez un e-mail et un mot de passe, puis choisissez Log In. L'e-mail et le mot de passe ne sont ni validés ni utilisés.
7. Dans le jeu Amazon GameLift Sample, choisissez Start. Le client du jeu recherche une session de jeu. S'il ne trouve pas de session, il en crée une. Le client de jeu démarre ensuite la session de jeu. Vous pouvez voir l'activité du jeu dans les journaux.

Exemples de journaux de serveurs de jeux

```
...
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
```

Exemples de journaux Amazon GameLift Local

```
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"
```

Arrêter le processus du serveur

> **Note**
>
> Cette rubrique fait référence au GameLift plugin Amazon pour Unity version 1.0.0, qui utilise le SDK de serveur 4.x ou une version antérieure.

Une fois que vous avez terminé votre exemple de jeu, arrêtez le serveur dans Unity.

1. Dans le client de jeu, choisissez Quitter ou fermez la fenêtre pour arrêter le client de jeu.
2. Dans Unity, dans la fenêtre de test local, choisissez Arrêter ou fermez les fenêtres du serveur de jeu pour arrêter le serveur.

Intégration de jeux avec le GameLift plugin Amazon pour Unreal Engine

Les rubriques de cette section décrivent le GameLift plugin Amazon pour Unreal Engine (UE) et expliquent comment l'utiliser pour préparer votre projet de jeu multijoueur en vue de son hébergement sur Amazon GameLift. Travaillez entièrement dans votre environnement de développement UE grâce aux flux de travail guidés du plugin pour répondre aux exigences de base de l'hébergement avec Amazon GameLift.

Amazon GameLift est un service entièrement géré qui permet aux développeurs de jeux de gérer et de faire évoluer des serveurs de jeux dédiés pour les jeux multijoueurs basés sur des sessions. Pour plus d'informations sur l' GameLift hébergement Amazon, consultezComment GameLift fonctionne Amazon.

Rubriques

- À propos du plugin
- Flux de travail du plugin
- Installez le plugin pour Unreal
- Configuration d'un profil AWS utilisateur
- Configurez votre jeu pour le tester avec Amazon GameLift Anywhere
- Déployez votre jeu sur un hébergement cloud avec des flottes EC2 gérées

À propos du plugin

Le plugin ajoute GameLift des outils et des fonctionnalités Amazon à l'éditeur UE. Les flux de travail guidés du plugin permettent d' GameLift intégrer Amazon à votre projet de jeu, de désigner un poste de travail comme hôte local pour les tests et de déployer le serveur de jeu sur l'hébergement GameLift cloud Amazon.

Utilisez les solutions d'hébergement prédéfinies du plugin pour déployer votre jeu. Configurez une flotte Amazon GameLift Anywhere avec votre poste de travail local comme hôte. Pour l'hébergement dans le cloud, choisissez entre deux scénarios de déploiement courants qui équilibrent de différentes manières la latence des joueurs, la disponibilité des sessions de jeu et les coûts. Un scénario inclut un système de FlexMatch matchmaking simple et un ensemble de règles. Utilisez ces solutions pour démarrer rapidement avec une structure d'hébergement prête pour la production en place, puis optimisez et personnalisez selon les besoins.

Le plugin inclut les composants suivants :

- Modules de plug-in pour l'éditeur UE. Lorsque le plugin est installé, un nouveau bouton du menu principal vous permet d'accéder aux GameLift fonctionnalités d'Amazon.
- bibliothèques C++ pour l'API du GameLift service Amazon avec fonctionnalités côté client.
- Bibliothèques Unreal pour le SDK GameLift du serveur Amazon (version 5).
- Contenu destiné aux tests, notamment une carte de jeu de démarrage et deux cartes de test avec des plans de base et des éléments d'interface utilisateur à utiliser pour tester l'intégration d'un serveur.
- Configurations modifiables, sous forme de AWS CloudFormation modèles, que le plugin utilise lors du déploiement de votre serveur de jeu pour l'hébergement.

Flux de travail du plugin

Les étapes suivantes décrivent une approche typique d'intégration et de déploiement d'un projet de jeu avec le GameLift plugin Amazon pour Unreal Engine. Vous devez effectuer ces étapes en utilisant l'éditeur UE et votre code de jeu.

1. Créez un profil utilisateur lié à votre AWS compte et fournissant des informations d'accès aux utilisateurs valides autorisés à utiliser Amazon GameLift.
2. Ajoutez du code serveur à votre projet de jeu pour établir la communication entre un serveur de jeu actif et le GameLift service with Amazon.
3. Ajoutez un code client à votre projet de jeu qui permet aux clients du jeu d'envoyer des demandes GameLift à Amazon pour démarrer de nouvelles sessions de jeu, puis de s'y connecter.
4. Utilisez le flux de travail Anywhere pour configurer votre station de travail locale en tant qu'hôte Anywhere pour votre serveur de jeu. Lancez votre serveur de jeu et votre client localement via le plugin, connectez-vous à une session de jeu et testez votre intégration.

5. Utilisez le flux de travail d'hébergement EC2 pour télécharger votre serveur de jeu intégré et déployer une solution d'hébergement cloud. Lorsque votre serveur de jeu est prêt, lancez votre client de jeu localement via le plugin, connectez-vous à une session de jeu et jouez au jeu.

Lorsque vous travaillez dans le plugin, vous allez créer et utiliser AWS des ressources. Ces actions peuvent entraîner des frais pour le AWS compte utilisé. Si vous êtes nouveau dans ce AWS domaine, ces actions peuvent être couvertes par le niveau AWS gratuit.

Installez le plugin pour Unreal

Cette section décrit les tâches d'installation initiales pour ajouter le plugin à un projet Unreal Engine. La fonctionnalité du plugin est disponible lorsque le projet est ouvert dans l'éditeur Unreal.

> **Note**
>
> Vous pouvez utiliser le GameLift plugin Amazon avec une version standard de l'éditeur UE, mais vous devez utiliser une version créée par le code source lorsque vous créez un package pour le build de votre serveur de jeu.

Avant de commencer

Voici ce dont vous avez besoin pour utiliser le GameLift plugin Amazon pour Unreal Engine :

- Package de lancement du GameLift plugin Amazon pour Unreal Engine. [Site de téléchargement].
- Microsoft Visual Studio 2019 ou version ultérieure.
- Une version créée par le code source de l'éditeur Unreal Engine. Vous avez besoin d'une version créée par le code source pour empaqueter les composants du serveur pour un jeu multijoueur. Pour plus de détails, y compris les prérequis supplémentaires, consultez la documentation d'Unreal Engine :
  - Accès au code source d'Unreal Engine sur GitHub You'll need GitHub et sur les comptes Epic Games.
  - Tutoriel de création d'Unreal Engine à partir des sources.
- Un projet de jeu multijoueur avec du code de jeu en C++. Si vous travaillez sur un projet Blueprint, consultez la documentation Unreal pour savoir comment générer du code source C++ pour votre projet.

Ajoutez le plugin à votre projet de jeu

Effectuez les tâches suivantes pour ajouter le plugin à votre projet de jeu.

Créez le SDK du serveur Amazon GameLift C++

1. Décompressez le package de publication du GameLift plugin Amazon pour Unreal Engine pour extraire deux fichiers zip :

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

   Décompressez ces fichiers.
2. Ouvrez le `GameLift-Cpp-ServerSDK-<>` dossier, puis suivez les instructions suivantes pour votre plate-forme : Linux ou Microsoft Windows.

Linux

1. Exécutez les commandes suivantes :

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

   Ces commandes créent le `/lib/aws-cpp-sdk-gamelift-server.so` fichier.
2. `/lib/aws-cpp-sdk-gamelift-server.so`Copiez dans le `amazon-gamelift-plugin-unreal/GameLiftPlugin/Source/GameliftServer/ThirdParty/GameLiftServerSDK/Linux/x86_64-unknown-linux-gnu/` répertoire.

Microsoft Windows

1. Exécutez les commandes suivantes :

   ```
   mkdir out
   cd out
   cmake -G "Visual Studio 17 2022" -DBUILD_FOR_UNREAL=1 ..
   ```

```
msbuild ALL_BUILD.vcxproj /p:Configuration=Release
```

Ces commandes créent les fichiers binaires suivants.

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

2. Copiez les fichiers dans le `amazon-gamelift-plugin-unreal\GameLiftPlugin\Source\GameliftServer\ThirdParty\GameLiftServerSDK\Win64\` répertoire.

Effectuez les tâches suivantes en travaillant sur les fichiers de votre projet de jeu.

1. Installez les fichiers du plugin.

   a. Localisez le dossier racine de votre projet de jeu, tel que`... > Unreal Projects/[project-name]/`. Si le dossier Plugins n'existe pas, créez-le.

   b. Accédez au `amazon-gamelift-plugin-unreal` dossier à partir duquel vous avez été décompressé. `amazon-gamelift-plugin-unreal-<>-sdk-<>.zip` Copiez le `GameLiftPlugin` dossier du `gamelift-plugin-unreal` dossier vers le `Plugins` dossier du répertoire du projet de jeu.

2. Ajoutez le plugin au **`.uproject`** fichier.

   a. Dans le dossier racine de votre projet de jeu, ouvrez le `.uproject` fichier.

   b. Mettez à jour le fichier pour ajouter GameLiftPlugin « » et WebBrowserWidget « » à la `Plugins` section et activez-les. Le code suivant montre le `.uproject` fichier mis à jour pour un jeu appelé « MyGame ».

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

```
    },
    {
      "Name": "WebBrowserWidget",
      "Enabled": true
    }
  ]
}
```

3. Modifiez la version de l'éditeur UE pour votre projet.

   Si vous avez créé un projet pour une version d'éditeur et que vous souhaitez maintenant passer à une autre version (telle qu'une version source), vous devez mettre à jour le projet.

   Dans le dossier racine de votre projet de jeu, sélectionnez le `.uproject` fichier et choisissez l'option Changer de version d'Unreal Engine. Sélectionnez une nouvelle version de l'éditeur.

4. Reconstruisez la solution du projet avec vos mises à jour.

   a. Dans le dossier racine du projet, recherchez un fichier de solution (`*.sln`). S'il n'en existe aucun, sélectionnez le `.uproject` fichier et choisissez l'option Générer les fichiers de projet Visual Studio.

   b. Ouvrez le fichier de solution et créez ou reconstruisez le projet.

5. Vérifiez que le plugin est activé dans l'éditeur UE.

   > **Note**
   >
   > Si vous avez déjà ouvert l'éditeur, vous devrez peut-être le redémarrer avant qu'il ne reconnaisse le nouveau plugin.

   a. Ouvrez le projet dans l'éditeur UE de votre choix.

   b. Consultez la barre d'outils principale de l'éditeur pour voir le nouveau bouton GameLift du menu Amazon [image requise].

   c. Recherchez les ressources du GameLift plugin Amazon dans le navigateur de contenu. Assurez-vous que l'option Afficher le contenu du plug-in est sélectionnée dans votre paramètre Options d'affichage.

Configuration d'un profil AWS utilisateur

Après avoir installé le plugin, configurez un profil et associez-le à un AWS compte utilisateur valide. Vous pouvez gérer plusieurs profils, mais vous ne pouvez avoir qu'un seul profil actif à la fois. Chaque fois que vous travaillez dans le plugin, sélectionnez un profil à utiliser.

La gestion de plusieurs profils vous permet de passer d'un scénario d'hébergement à un autre. Par exemple, vous pouvez configurer des profils avec les mêmes AWS informations d'identification mais avec des AWS régions différentes. Vous pouvez également configurer des profils avec différents AWS comptes ou avec différents utilisateurs/ensembles d'autorisations.

> **Note**
>
> Si vous avez installé la AWS CLI sur votre poste de travail et qu'un profil est déjà configuré, le GameLift plugin Amazon peut le détecter et le répertoriera comme profil existant. Le plugin sélectionne automatiquement n'importe quel profil nommé`[default]`. Vous pouvez utiliser un profil existant ou en créer un nouveau.

Pour gérer vos AWS profils

1. Dans la barre d'outils principale de l'éditeur Unreal, choisissez le GameLift menu Amazon, puis sélectionnez Set AWS User Profiles. Cette action ouvre les paramètres du projet pour le plugin. Développez la section Profils AWS utilisateurs.
2. Si le plugin ne détecte aucun profil existant, il vous invite à en créer un. Vous pouvez créer un nouveau profil à l'aide d'un compte nouveau ou d'un AWS compte existant.

   > **Note**
   >
   > Vous devez utiliser la console AWS de gestion pour créer un nouveau AWS compte et créer ou mettre à jour un utilisateur doté des autorisations appropriées.

   Lorsque vous configurez un profil, vous avez besoin des informations suivantes :

   - Un compte AWS. Si vous devez créer un nouveau AWS compte, suivez les instructions pour le créer. Voir Créer un AWS compte pour plus de détails.

- Un AWS utilisateur autorisé à utiliser Amazon GameLift et les autres AWS services requis. Consultez les instructions Configurez un Compte AWS relatives à la configuration d'un utilisateur AWS Identity and Access Management (IAM) avec des GameLift autorisations Amazon et un accès programmatique avec des informations d'identification à long terme.
- Informations d'identification de votre AWS utilisateur. Ces informations d'identification se composent d'un identifiant de clé d'AWSaccès et d'une clé AWS secrète. Consultez Obtenir vos clés d'accès pour plus de détails.
- AWSrégion. Il s'agit d'un emplacement géographique dans lequel vous souhaitez créer vos AWS ressources pour l'hébergement. Pendant le développement, nous vous recommandons d'utiliser une région proche de votre emplacement physique. Consultez la liste des AWSrégions prises en charge.

3. Si le plugin détecte un profil existant, vous n'êtes pas invité à en créer un. Si vous souhaitez mettre à jour un profil ou en créer un nouveau, choisissez Gérer votre profil.

Pour démarrer votre profil, procédez comme suit :

Tous les profils doivent être amorcés pour être utilisés avec le plugin Amazon GameLift . Le bootstrapping crée un compartiment Amazon S3 spécifique au profil. Il est utilisé pour stocker les configurations de projet, créer des artefacts et d'autres dépendances. Les buckets ne sont pas partagés entre d'autres profils.

Le bootstrapping implique la création de nouvelles AWS ressources et peut entraîner des coûts.

1. Dans la barre d'outils principale de l'éditeur Unreal, choisissez l' GameLift icône Amazon, puis sélectionnez Définir les profils AWS utilisateur. Cette action ouvre les paramètres du projet pour le plugin. Développez la section Profils AWS utilisateurs.
2. Dans la section Bootstrap your profile, sélectionnez un profil dans la liste déroulante et vérifiez l'état du bootstrap. Si le statut indique qu'aucun compartiment n'existe, cliquez sur le bouton Bootstrap et créez un profil pour créer un compartiment Amazon S3 pour le profil sélectionné.
3. Attendez que le statut du bootstrap passe à « Actif ». Cette opération peut prendre quelques minutes.

Configurez votre jeu pour le tester avec Amazon GameLift Anywhere

Dans ce flux de travail, vous ajoutez du code de jeu client et serveur pour les GameLift fonctionnalités Amazon, et vous utilisez le plugin pour désigner votre station de travail locale comme hôte de serveur

de jeu de test. Lorsque vous avez terminé les tâches d'intégration, utilisez le plugin pour créer les composants du client et du serveur de votre jeu.

Pour démarrer le flux de travail Amazon GameLift Anywhere :

- Dans la barre d'outils principale de l'éditeur Unreal, choisissez le GameLift menu Amazon, puis sélectionnez Host with Anywhere. Cette action ouvre la page du plugin Deploy Anywhere, qui présente un processus en six étapes pour intégrer, créer et lancer les composants de votre jeu.

Étape 1 : Définissez votre profil

Choisissez le profil que vous souhaitez utiliser lorsque vous suivez ce flux de travail. Le profil que vous sélectionnez a un impact sur toutes les étapes du flux de travail. Toutes les ressources que vous créez sont associées au AWS compte du profil et sont placées dans la AWS région par défaut du profil. Les autorisations de l'utilisateur du profil déterminent votre accès aux AWS ressources et aux actions.

1. Sélectionnez un profil dans la liste déroulante des profils disponibles. Si vous n'avez pas encore de profil ou si vous souhaitez en créer un nouveau, rendez-vous dans le GameLift menu Amazon et choisissez Set AWS User Profiles.
2. Si le statut du bootstrap n'est pas « Actif », choisissez le profil Bootstrap et attendez que le statut passe à « Actif ».

Étape 2 : configurez votre code de jeu

Au cours de cette étape, vous apportez une série de mises à jour à votre code client et serveur pour ajouter des fonctionnalités d'hébergement. Si vous n'avez pas encore configuré de version source de l'éditeur Unreal, le plugin fournit des liens vers les instructions et le code source.

Avec le plugin, vous pouvez profiter de certaines commodités lors de l'intégration de votre code de jeu. Vous pouvez effectuer une intégration minimale pour configurer les fonctionnalités d'hébergement de base. Vous pouvez également effectuer une intégration personnalisée plus poussée. Les informations de cette section décrivent l'option d'intégration minimale. Utilisez les cartes de test incluses dans le plugin pour ajouter la GameLift fonctionnalité client Amazon à votre projet de jeu. Pour l'intégration au serveur, utilisez l'exemple de code fourni pour mettre à jour le mode de jeu de votre projet.

Intégrez le mode de jeu de votre serveur

Ajoutez un code serveur à votre jeu pour permettre la communication entre votre serveur de jeu et le GameLift service Amazon. Votre serveur de jeu doit être en mesure de répondre aux demandes d'Amazon GameLift, par exemple pour démarrer une nouvelle session de jeu, et également de signaler l'état de santé du serveur de jeu et les connexions des joueurs.

1. Dans votre éditeur de code, ouvrez le fichier solution (`.sln`) pour votre projet de jeu, qui se trouve généralement dans le dossier racine du projet. Par exemple : `GameLiftUnrealApp.sln`.
2. La solution étant ouverte, localisez le fichier d'en-tête du mode jeu du projet : `[project-name]GameMode.h` file. Par exemple : `GameLiftUnrealAppGameMode.h`.
3. Modifiez le fichier d'en-tête pour l'aligner sur l'exemple de code suivant. Assurez-vous de remplacer « GameLiftServer » par le nom de votre propre projet. Ces mises à jour sont spécifiques au serveur de jeu ; nous vous recommandons de créer une copie de sauvegarde des fichiers du mode de jeu d'origine pour les utiliser avec votre client.

```
// 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. Ouvrez le `[project-name]GameMode.cpp` fichier source correspondant (par exemple`GameLiftUnrealAppGameMode.cpp`). Modifiez le code pour l'aligner sur l'exemple de code suivant. Assurez-vous de remplacer « GameLiftUnrealApp » par le nom de votre propre projet. Ces mises à jour sont spécifiques au serveur de jeu ; nous vous recommandons de créer une copie de sauvegarde du fichier original pour l'utiliser avec votre client.

   L'exemple de code suivant montre comment ajouter les éléments minimaux requis pour l'intégration du serveur à Amazon GameLift :

   - Initialisez un client d' GameLift API Amazon. L'`InitSDK()`appel avec les paramètres du serveur est obligatoire pour une flotte Amazon GameLift Anywhere. Lorsque vous vous connectez à une flotte Anywhere, le plugin stocke les paramètres du serveur sous forme d'arguments de console. L'exemple de code permet d'accéder aux valeurs lors de l'exécution.
   - Implémentez les fonctions de rappel requises pour répondre aux demandes du GameLift service Amazon, notamment `OnStartGameSession``OnProcessTerminate`, et`onHealthCheck`.
   - Appelez `ProcessReady()` un port désigné pour informer le GameLift service Amazon lorsque vous êtes prêt à héberger des sessions de jeu.

```
// 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
}
```

Intégrez la carte de jeu de votre client

La carte du jeu de démarrage contient une logique de plan et des éléments d'interface utilisateur qui incluent déjà un code de base pour demander des sessions de jeu et utiliser les informations de connexion pour se connecter à une session de jeu. Vous pouvez utiliser la carte telle quelle ou les modifier selon vos besoins. Utilisez la carte du jeu de démarrage avec d'autres éléments du jeu, tels que le projet de modèle à la troisième personne fourni par Unreal Engine. Ces ressources sont disponibles dans le navigateur de contenu. Vous pouvez les utiliser pour tester les flux de travail de déploiement du plugin ou comme guide pour créer un service de backend personnalisé pour votre jeu.

La carte de démarrage présente les caractéristiques suivantes :

- Il inclut une logique à la fois pour une flotte Anywhere et une flotte EC2 gérée. Lorsque vous gérez votre client, vous pouvez choisir de vous connecter à l'une ou l'autre des flottes.
- Les fonctionnalités du client incluent la recherche d'une session de jeu SearchGameSessions (()), la création d'une nouvelle session de jeu CreateGameSession (()) et la participation directe à une session de jeu.

- Il obtient un identifiant de joueur unique à partir du pool d'utilisateurs Amazon Cognito de votre projet (cela fait partie d'une solution Anywhere déployée).

Pour utiliser la carte du jeu de démarrage

1. Dans l'éditeur UE, ouvrez la page Paramètres du projet, cartes et modes, puis développez la section Cartes par défaut.
2. Pour Editor Startup Map, sélectionnez StartupMap « » dans la liste déroulante. Vous devrez peut-être rechercher le fichier, qui se trouve dans`... > Unreal Projects/[project-name]/Plugins/Amazon GameLift Plugin Content/Maps`.
3. Pour la carte par défaut du jeu, sélectionnez le même StartupMap « » dans la liste déroulante.
4. Pour la carte par défaut du serveur, sélectionnez « ThirdPersonMap ». Il s'agit d'une carte par défaut incluse dans votre projet de jeu. Cette carte est conçue pour deux joueurs.
5. Ouvrez le panneau de détails de la carte par défaut du serveur. Définissez GameMode Override sur « Aucun ».
6. Développez la section Modes par défaut et définissez le mode de jeu par défaut global sur le mode de jeu que vous avez mis à jour pour l'intégration de votre serveur.

Après avoir apporté ces modifications à votre projet, vous êtes prêt à créer les composants de votre jeu.

Construisez les composants de votre jeu

1. Création de nouveaux fichiers cibles pour le serveur et le client

    a. Dans le dossier de votre projet de jeu, allez dans le dossier Source et recherchez les `Target.cs` fichiers.

    b. Copiez le fichier `[project-name]Editor.Target.cs` dans deux nouveaux fichiers nommés `[project-name]Client.Target.cs` et`[project-name]Server.Target.cs`.

    c. Modifiez chacun des nouveaux fichiers pour mettre à jour le nom de classe et les valeurs du type de cible, comme indiqué :

```
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. Mettez à jour le `.Build.cs` fichier.

   a. Ouvrez le `.Build.cs` fichier correspondant à votre projet. Ce fichier se trouve dans le dossier `UnrealProjects/[project name]/Source/[project name]/[project name].Build.cs`.

   b. Mettez à jour la `ModuleRules` classe comme indiqué dans l'exemple de code suivant.

```
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. Reconstruisez la solution de votre projet de jeu.
4. Ouvrez votre projet de jeu dans une version source de l'éditeur Unreal Engine.
5. Procédez comme suit pour votre client et votre serveur :

   a. Choisissez une cible. Accédez à Plateformes, Windows et sélectionnez l'une des options suivantes :

      - Serveur : `[your-application-name]Server`
      - Client :`[your-application-name]Client`

   b. Démarrez la construction. Accédez à Platform, Windows, Package Project.

Chaque processus d'empaquetage génère un exécutable : `[your-application-name]Client.exe` ou`[your-application-name]Server.exe`.

Dans le plugin, définissez les chemins d'accès aux exécutables de génération du client et du serveur sur votre poste de travail local.

Étape 3 : Connectez-vous à une flotte n'importe où

Au cours de cette étape, vous désignez une flotte Anywhere à utiliser. Une flotte Anywhere définit un ensemble de ressources informatiques, qui peuvent être situées n'importe où, pour l'hébergement de serveurs de jeux.

- Si le AWS compte que vous utilisez actuellement possède des flottes Anywhere existantes, ouvrez le champ déroulant Nom de la flotte et choisissez une flotte. Cette liste déroulante affiche uniquement les flottes Anywhere de la AWS région correspondant au profil utilisateur actuellement actif.
- S'il n'existe aucune flotte existante, ou si vous souhaitez en créer une nouvelle, choisissez Create new Anywhere fleet et saisissez un nom de flotte.

Une fois que vous avez choisi une flotte Anywhere pour votre projet, Amazon GameLift vérifie que l'état de la flotte est actif et affiche l'identifiant de la flotte. Vous pouvez suivre la progression de cette demande dans le journal de sortie de l'éditeur Unreal.

Étape 4 : Enregistrez votre poste de travail

Au cours de cette étape, vous enregistrez votre station de travail locale en tant que ressource de calcul dans le nouveau parc Anywhere.

1. Entrez un nom de calcul pour votre machine locale. Si vous ajoutez plusieurs ordinateurs dans le parc, les noms doivent être uniques.
2. Fournissez une adresse IP pour votre machine locale. Ce champ correspond par défaut à l'adresse IP publique de votre machine. Vous pouvez également utiliser localhost (127.0.0.1) tant que vous exécutez votre client de jeu et votre serveur sur la même machine.
3. Choisissez Register compute. Vous pouvez suivre la progression de cette demande dans le journal de sortie de l'éditeur Unreal.

En réponse à cette action, Amazon GameLift vérifie qu'il peut se connecter au calcul et renvoie des informations sur le calcul nouvellement enregistré. Il crée également les arguments de console dont les exécutables de vos jeux ont besoin lors de l'initialisation de la communication avec le service Amazon. GameLift

Étape 5 : générer un jeton d'authentification

Les processus du serveur de jeu qui s'exécutent sur votre ordinateur Anywhere ont besoin d'un jeton d'authentification pour appeler le GameLift service. Le plugin génère et stocke automatiquement un jeton d'authentification pour la flotte Anywhere chaque fois que vous lancez le serveur de jeu depuis le plugin. La valeur du jeton d'authentification est stockée sous forme d'argument de ligne de commande, que le code de votre serveur peut récupérer lors de l'exécution.

Vous n'avez aucune action à effectuer au cours de cette étape.

Étape 6 : Lancer le jeu

À ce stade, vous avez terminé toutes les tâches nécessaires pour lancer et jouer à votre jeu multijoueur sur un poste de travail local à l'aide d'Amazon GameLift.

1. Lancez votre serveur de jeu. Le serveur de jeu informera Amazon GameLift lorsqu'il sera prêt à héberger des sessions de jeu.
2. Lancez votre client de jeu et utilisez les nouvelles fonctionnalités pour démarrer une nouvelle session de jeu. Cette demande est envoyée à Amazon GameLift via le nouveau service principal. En réponse GameLift, Amazon appelle le serveur de jeu, qui fonctionne sur votre machine locale, pour démarrer une nouvelle session de jeu. Lorsque la session de jeu est prête à accepter des joueurs, Amazon GameLift fournit les informations de connexion permettant au client du jeu de rejoindre la session de jeu.

Déployez votre jeu sur un hébergement cloud avec des flottes EC2 gérées

Dans ce flux de travail, vous utilisez le plugin pour modifier votre jeu afin de l'héberger sur des ressources informatiques basées sur le cloud gérées par Amazon GameLift. Vous ajoutez le code de jeu client et serveur pour les GameLift fonctionnalités Amazon, puis vous téléchargez la version de votre serveur sur le GameLift service Amazon pour le déployer sur les ressources basées sur le cloud. Lorsque ce flux de travail sera terminé, vous disposerez d'un client de jeu fonctionnel qui pourra se connecter à vos serveurs de jeu dans le cloud.

Pour démarrer le flux de travail Amazon EC2 GameLift géré par Amazon :

- Dans la barre d'outils principale de l'éditeur Unreal, choisissez le GameLift menu Amazon, puis sélectionnez Host with Managed EC2. Cette action ouvre la page du plugin Deploy Amazon EC2 Fleet, qui présente un processus en six étapes pour intégrer, créer, déployer et lancer les composants de votre jeu.

Étape 1 : Définissez votre profil

Choisissez le profil que vous souhaitez utiliser lorsque vous suivez ce flux de travail. Le profil que vous sélectionnez a un impact sur toutes les étapes du flux de travail. Toutes les ressources que vous créez sont associées au AWS compte du profil et sont placées dans la AWS région par défaut du profil. Les autorisations de l'utilisateur du profil déterminent votre accès aux AWS ressources et aux actions.

1. Sélectionnez un profil dans la liste déroulante des profils disponibles. Si vous n'avez pas encore de profil ou si vous souhaitez en créer un nouveau, rendez-vous dans le GameLift menu Amazon et choisissez Set AWS User Profiles.
2. Si le statut du bootstrap n'est pas « Actif », choisissez le profil Bootstrap et attendez que le statut passe à « Actif ».

Étape 2 : configurez votre code de jeu

Au cours de cette étape, vous apportez une série de mises à jour à votre code client et serveur pour ajouter des fonctionnalités d'hébergement. Si vous n'avez pas encore configuré de version source de l'éditeur Unreal, le plugin fournit des liens vers les instructions et le code source.

Si vous avez intégré votre jeu pour l'utiliser dans une flotte Anywhere, vous n'avez pas besoin de modifier votre code de jeu. Si vous utilisez la carte du jeu de démarrage, elle fonctionne également avec les déploiements EC2.

- Configurez votre code de jeu (n'importe où)
- Construisez les composants de votre jeu

Après avoir créé votre serveur de jeu, effectuez les tâches suivantes pour le préparer au téléchargement sur Amazon GameLift.

Pour empaqueter la version de votre serveur pour le déploiement dans le cloud

Dans le `WindowsServer` dossier, où l'éditeur Unreal empaquète les fichiers de build de votre serveur par défaut, effectuez les ajouts suivants

1. Copiez le script d'installation, inclus dans le téléchargement du plugin, à la racine du `WindowsServer` dossier. Cherchez le dossier`[project-name]/Plugins/Resources/CloudFormation/extra_server_resources/install.bat`. Amazon GameLift utilise ce fichier pour installer la version du serveur sur chaque ressource d'hébergement EC2.
2. Copiez le `VC_redist.x64.exe` fichier, inclus dans votre installation de Visual Studio, à la racine du `WindowsServer` dossier. Ce fichier se trouve généralement à l'adresse`C:/Program Files (x86)/Microsoft Visual Studio/2019/Professional/VC/Redist/MSVC/v142`.

3. Copiez les DLL OpenSSL pour la version intégrée de votre serveur de jeu dans le dossier. `WindowsServer/MyGame/Binaries/Win64` Assurez-vous que les DLL correspondent à la version utilisée dans la version du serveur. Copiez les fichiers suivants :

   - `libssl-3-x64.dll`
   - `libcrypto-3-x64.dll`

Étape 3 : Sélectionnez le scénario de déploiement

Au cours de cette étape, vous choisissez la solution d'hébergement de jeux que vous souhaitez déployer à ce stade. Vous pouvez effectuer plusieurs déploiements de votre jeu, en utilisant n'importe quel scénario.

- Flotte à région unique : déploie votre serveur de jeu sur une flotte unique de ressources d'hébergement dans la région par défaut AWS du profil actif. Ce scénario constitue un bon point de départ pour tester l'intégration de votre serveur AWS et la configuration de la version du serveur. Il déploie les ressources suivantes :
  - AWSflotte (à la demande) avec la version de votre serveur de jeu installée et en cours d'exécution.
  - Groupe d'utilisateurs et client Amazon Cognito pour permettre aux joueurs de s'authentifier et de démarrer une partie.
  - Autorisateur de passerelle d'API qui relie le groupe d'utilisateurs aux API.
  - WebACL pour limiter les appels excessifs des joueurs à la passerelle API.
  - Passerelle API + fonction Lambda permettant aux joueurs de demander une machine à sous. Cette fonction appelle `CreateGameSession()` si aucune fonction n'est disponible.
  - Passerelle API + fonction Lambda permettant aux joueurs d'obtenir des informations de connexion pour leur demande de jeu.
- FlexMatch flotte : déploie votre serveur de jeu sur un ensemble de flottes et met en place un FlexMatch système de matchmaking avec des règles pour créer des parties entre joueurs. Ce scénario utilise un hébergement Spot à faible coût avec une structure multi-flottes et multi-sites pour une disponibilité durable. Cette approche est utile lorsque vous êtes prêt à commencer à concevoir un composant de matchmaking pour votre solution d'hébergement. Dans ce scénario, vous allez créer les ressources de base pour cette solution, que vous pourrez personnaliser ultérieurement selon vos besoins. Il déploie les ressources suivantes :

- FlexMatch configuration du matchmaking et règles de matchmaking définies pour accepter les demandes des joueurs et former des matchs.
- Trois AWS flottes équipées de votre version de serveur de jeu sont installées et fonctionnent à plusieurs endroits. Comprend deux flottes Spot et une flotte à la demande en tant que sauvegarde.
- AWSfile d'attente de placement de sessions de jeu qui répond aux demandes de matchs proposés en trouvant la meilleure ressource d'hébergement possible (en fonction de la viabilité, du coût, de la latence des joueurs, etc.) et en démarrant une session de jeu.
- Groupe d'utilisateurs et client Amazon Cognito pour permettre aux joueurs de s'authentifier et de démarrer une partie.
- Autorisateur de passerelle d'API qui relie le groupe d'utilisateurs aux API.
- WebACL pour limiter les appels excessifs des joueurs à la passerelle API.
- Passerelle API + fonction Lambda permettant aux joueurs de demander une machine à sous. Cette fonction appelle`StartMatchmaking()`.
- Passerelle API + fonction Lambda permettant aux joueurs d'obtenir des informations de connexion pour leur demande de jeu.
- Tables Amazon DynamoDB pour stocker les tickets de matchmaking pour les joueurs et les informations sur les sessions de jeu.
- Rubrique SNS + Fonction Lambda pour GameSessionQueue gérer les événements.

Étape 4 : définir les paramètres du jeu

Dans cette étape, vous décrivez le jeu vers lequel vous souhaitez le téléchargerAWS.

- Nom de version du serveur : Donnez un nom significatif à la version de votre serveur de jeu. AWSutilise ce nom pour faire référence à la copie de la version de votre serveur qui est téléchargée et utilisée pour les déploiements.
- Système d'exploitation pour la version du serveur : entrez le système d'exploitation sur lequel votre serveur est conçu pour fonctionner. Cela indique le AWS type de ressources informatiques à utiliser pour héberger votre jeu.
- Dossier du serveur de jeu : identifiez le chemin d'accès au dossier de compilation de votre serveur local.
- Version du serveur de jeu : identifiez le chemin d'accès au fichier exécutable du serveur de jeu.
- Chemin du client de jeu : identifiez le chemin d'accès à l'exécutable du client de jeu.

- Résultat de configuration du client : ce champ doit pointer vers un dossier de votre build client contenant votre AWS configuration. Recherchez-le à l'endroit suivant :`[client-build]/[project-name]/Content/CloudFormation`.

Étape 5 : scénario de déploiement

Au cours de cette étape, vous déployez votre jeu sur une solution d'hébergement cloud en fonction du scénario de déploiement que vous avez choisi. Ce processus peut prendre jusqu'à 40 minutes pour AWS valider la version de votre serveur, approvisionner les ressources d'hébergement, installer votre serveur de jeu, lancer les processus du serveur et les préparer à héberger des sessions de jeu.

Pour démarrer le déploiement, choisissez Deploy CloudFormation. Vous pouvez suivre l'état de votre hébergement de jeux ici. Pour obtenir des informations plus détaillées, vous pouvez vous connecter à la console AWS de gestion AWS et consulter les notifications d'événements. Assurez-vous de vous connecter en utilisant le même compte, le même utilisateur et AWS la même région que le profil utilisateur actif dans le plugin.

Lorsque le déploiement est terminé, votre serveur de jeu est installé sur une instance AWS EC2. Au moins un processus serveur est en cours d'exécution et prêt à démarrer une session de jeu.

Étape 6 : Lancer le client

À ce stade, vous avez terminé toutes les tâches nécessaires pour lancer et jouer à votre jeu multijoueur hébergé par Amazon GameLift. Pour jouer au jeu, lancez une instance de votre client de jeu.

Si vous avez déployé le scénario de flotte unique, vous pouvez ouvrir une instance client unique avec un seul joueur, accéder à la carte du serveur et vous déplacer. Ouvrez des instances supplémentaires du client de jeu pour ajouter un deuxième joueur à la même carte de jeu du serveur.

Si vous avez déployé le FlexMatch scénario, la solution attend qu'au moins deux clients soient mis en file d'attente pour le placement des sessions de jeu avant que les joueurs puissent accéder à la carte du serveur.

Obtenir des données de flotte pour une GameLift instance Amazon

Dans certains cas, votre version de jeu personnalisée ou le script de vos serveurs en temps réel peuvent nécessiter des informations sur la GameLift flotte Amazon. Par exemple, la version ou le script de votre jeu peut inclure du code pour :

- Surveillez l'activité en fonction des données de la flotte.
- Regroupez les indicateurs pour suivre l'activité en fonction des données de flotte. (De nombreux jeux utilisent ces données pour LiveOps des activités.)
- Fournissez des données pertinentes aux services de jeu personnalisés, par exemple pour le matchmaking, l'augmentation des capacités ou les tests.

Les informations sur le parc sont disponibles sous forme de fichier JSON sur chaque instance aux emplacements suivants :

- Windows: `C:\GameMetadata\gamelift-metadata.json`
- Linux : `/local/gamemetadata/gamelift-metadata.json`

Le `gamelift-metadata.json` fichier inclut les attributs d'une ressource de GameLift flotte Amazon.

Exemple de fichier 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"
}
```

Ajouter le FlexMatch matchmaking

Utilisez Amazon GameLift FlexMatch pour ajouter une fonctionnalité de jumelage de joueurs à vos jeux GameLift hébergés sur Amazon. Vous pouvez l'utiliser FlexMatch avec des serveurs de jeu personnalisés ou des serveurs en temps réel.

FlexMatch associe le service de mise en relation avec un moteur de règles personnalisables. Vous déterminez comment associer les joueurs en fonction de leurs attributs et des modes de jeu adaptés à votre jeu. FlexMatchgère les détails de l'évaluation des joueurs qui recherchent un match, de la formation des matchs avec une ou plusieurs équipes et du démarrage des sessions de jeu pour accueillir les matchs.

Pour utiliser le FlexMatch service complet, vos ressources d'hébergement doivent être configurées avec des files d'attente. Amazon GameLift utilise des files d'attente pour localiser les meilleurs emplacements d'hébergement possibles pour les jeux dans plusieurs régions et différents types d'ordinateurs. Les GameLift files d'attente Amazon peuvent notamment utiliser les données de latence, lorsqu'elles sont fournies par les clients du jeu, pour placer des sessions de jeu afin que les joueurs bénéficient de la latence la plus faible possible lorsqu'ils jouent.

Pour plus d'informations sur l'FlexMatchinclusion d'une aide détaillée sur l'intégration du matchmaking dans vos jeux, consultez les rubriques suivantes du guide du GameLift FlexMatch développeur Amazon :

- Comment GameLift FlexMatch fonctionne Amazon
- FlexMatchétapes d'intégration

Gestion de l'hébergement avec les GameLift conteneurs Amazon

Cette documentation concerne une fonctionnalité en version préliminaire publique. Elle est susceptible d'être modifiée.

Amazon GameLift fournit un service complet d'hébergement dans le cloud pour prendre en charge les solutions conteneurisées pour l'hébergement de serveurs de jeux. Avec les flottes de GameLift conteneurs Amazon, vous pouvez tirer parti des avantages des conteneurs tels que la portabilité, l'agilité et la tolérance aux pannes.

Fonctions principales

Les fonctionnalités suivantes sont disponibles avec les flottes de GameLift conteneurs Amazon.

- Développez une architecture de conteneur personnalisée avec des conteneurs légers pour exécuter le logiciel de votre serveur de jeu sur les ressources GameLift d'hébergement Amazon.
- Incluez l' GameLift agent Amazon pour gérer le cycle de vie des processus du serveur de jeu dans vos conteneurs. L'agent informatique exécute vos instructions sur le moment et la manière de démarrer les processus du serveur et sur le nombre de processus à gérer pour l'hébergement des sessions de jeu.
- Personnalisez les ressources fournies par Amazon GameLift pour créer des images de conteneur avec votre application de serveur de jeu. Utilisez le fichier docker fourni pour créer une image de conteneur basée sur Linux. Stockez les images de vos flottes de conteneurs dans un référentiel privé Amazon Elastic Container Registry (Amazon ECR).
- Offrez des expériences aux joueurs à faible latence en déployant les ressources du parc de conteneurs dans n'importe quelle zone Région AWS ou zone locale prise GameLift en charge par Amazon. Créez des flottes de conteneurs sur plusieurs sites pour une gestion de flotte rationalisée. veuillez consulter Sites GameLift d'hébergement Amazon.
- Testez vos solutions d'hébergement de jeux conteneurisés avec une flotte Amazon GameLift Anywhere. Utilisez Anywhere pour tester localement le développement de votre solution, y compris votre intégration au GameLift SDK Amazon et les configurations de vos images de conteneur.
- Suivez les performances d'hébergement de jeux grâce à des indicateurs de performance spécifiques au conteneur. Surveillez l'état des ressources de votre flotte à l'aide de métriques matérielles.

- Utilisez les outils de placement de sessions de GameLift jeu d'Amazon, notamment les files d'attente et le FlexMatch matchmaking, pour associer les joueurs aux meilleures sessions de jeu possibles hébergées sur vos flottes de conteneurs.
- Gérez les ressources du parc de conteneurs à l'aide de AWS CloudFormation modèles pour Amazon GameLift.

Utilisation de flottes de conteneurs lors de la prévisualisation publique

La nouvelle fonctionnalité relative aux flottes de conteneurs est actuellement en version préliminaire publique. Au cours de cette phase, les GameLift fonctionnalités Amazon suivantes sont prises en charge :

- Utilisez des flottes de conteneurs pour héberger des serveurs de jeux conçus pour Linux. Les flottes de conteneurs utilisent `Amazon_Linux_2023` et prennent en charge les images de conteneurs Linux. Les conteneurs Windows ne sont pas pris en charge.
- Intégrez des projets de serveur de jeu uniquement avec le SDK Amazon GameLift Server version 5+. Les versions précédentes ne sont pas prises en charge.
- Utilisez l'un des types d'instances Amazon EC2 On-Demand pris en charge par Amazon GameLift . Les flottes Spot ne sont pas prises en charge pour le moment.

Comment fonctionnent les conteneurs sur Amazon GameLift

Cette documentation concerne une fonctionnalité en version préliminaire publique. Elle est susceptible d’être modifiée.

Les flottes de GameLift conteneurs Amazon sont conçues pour apporter de la flexibilité dans le déploiement et le dimensionnement de vos applications conteneurisées. Il utilise Amazon Elastic Container Service (Amazon ECS) pour gérer le déploiement et l'exécution des tâches pour vos flottes Amazon GameLift . Cette rubrique décrit les éléments structurels de base pour faire fonctionner des conteneurs sur un parc GameLift géré par Amazon, illustre les architectures courantes et décrit certains concepts de base.

Composants du parc de conteneurs

Flotte

Un parc de conteneurs est un ensemble d'instances Amazon EC2, gérées par Amazon GameLift, qui exécutent vos serveurs de jeux conteneurisés. Lorsque vous créez une flotte, vous configurez la manière dont votre architecture de conteneur et le logiciel de serveur de jeu sont déployés sur chaque instance de flotte. Vous pouvez déployer une flotte de conteneurs sur un Région AWS ou plusieurs sites géographiques. Vous pouvez utiliser les outils de dimensionnement GameLift manuel ou automatique d'Amazon pour augmenter la capacité d'une flotte de conteneurs afin d'accueillir des sessions de jeu et des joueurs.

Instance

Une instance Amazon EC2 est le serveur virtuel qui fournit la capacité de calcul pour l'hébergement de votre jeu. Avec Amazon GameLift, vous pouvez choisir parmi différents types d'instances. Chaque type d'instance offre une combinaison différente de CPU, de mémoire, de stockage et de capacité réseau.

Lorsque vous créez un parc de conteneurs, Amazon GameLift déploie des instances en fonction du type d'instance que vous avez choisi et de la configuration de votre flotte. Chaque instance de flotte déployée est identique et exécute le logiciel de votre serveur de jeu conteneurisé de la même manière. Le nombre d'instances d'une flotte détermine la taille de la flotte et sa capacité d'hébergement de jeux.

Groupe de conteneurs

Amazon GameLift utilise le concept de groupe de conteneurs pour décrire et gérer un ensemble de conteneurs. Un groupe de conteneurs est similaire à une « tâche » ou à un « pod » de conteneurs. Au sein de chaque groupe de conteneurs, vous pouvez définir la manière dont les conteneurs partagent les ressources de processeur et de mémoire disponibles. Vous pouvez également configurer des dépendances entre les conteneurs et gérer le cycle de vie du groupe de conteneurs.

Les groupes de conteneurs peuvent être répliqués sur chaque instance de flotte afin d'optimiser l'utilisation des ressources. Vous pouvez gérer la réplication en définissant la stratégie de planification d'un groupe de conteneurs, comme suit :

- Les groupes de répliques de conteneurs gèrent les conteneurs qui exécutent votre application de serveur de jeu et les logiciels associés. Toutes les flottes de conteneurs doivent définir un groupe de conteneurs répliques. Un groupe de répliques peut avoir plusieurs copies sur chaque

instance de flotte, en fonction des exigences du groupe de conteneurs et des ressources du type d'instance utilisé. Tous les conteneurs du groupe de répliques sont automatiquement redimensionnés sur une instance.

- Les groupes de conteneurs de démons, qui sont facultatifs, peuvent être utiles pour exécuter des services d'arrière-plan ou des programmes utilitaires, par exemple pour la surveillance. Le logiciel de votre serveur de jeu ne dépend pas directement des processus d'un groupe de démons. Les groupes de conteneurs de démons ne sont pas répliqués : chaque instance de flotte possède au plus une copie du groupe de démons. Cela signifie que les conteneurs d'un groupe de démons ne peuvent pas être répartis sur une instance de flotte au même titre que les conteneurs d'un groupe de répliques.

Une flotte de conteneurs doit disposer d'un groupe de conteneurs répliqués et peut éventuellement comporter un groupe de démons.

Conteneur

Le conteneur est l'élément le plus fondamental d'une architecture basée sur un conteneur. Il se compose d'une image de conteneur avec des exécutables logiciels et des fichiers dépendants. Lorsque vous définissez un conteneur à utiliser avec Amazon GameLift, vous configurez la manière dont le logiciel s'exécute dans le conteneur.

Chaque groupe de conteneurs d'une flotte de conteneurs doit avoir un conteneur désigné « essentiel ». Un conteneur essentiel conditionne le cycle de vie d'un groupe de conteneurs. Si le conteneur essentiel tombe en panne, l'ensemble du groupe de conteneurs redémarre.

Les types de contenants incluent :

- Essential Replica Container inclut tout ce dont vous avez besoin pour exécuter les processus de votre serveur de jeu et héberger des sessions de jeu pour les joueurs. Cela inclut la version de votre serveur de jeu, qui est intégrée au SDK GameLift du serveur Amazon, et les logiciels dépendants. Il inclut également l' GameLift agent Amazon, qui gère le cycle de vie des processus de votre serveur de jeu. Le groupe de répliques de conteneurs d'une flotte possède exactement une réplique de conteneur essentielle.
- Les conteneurs de répliques non essentiels, également appelés conteneurs « sidecar », exécutent des logiciels compatibles avec votre application de serveur de jeu. L'utilisation d'un conteneur annexe vous permet d'exécuter et de faire évoluer les logiciels de support parallèlement à vos serveurs de jeu, mais de les gérer comme des conteneurs distincts. Si ce type de conteneur échoue, seul le conteneur lui-même redémarre ; le groupe de conteneurs n'est pas impacté.

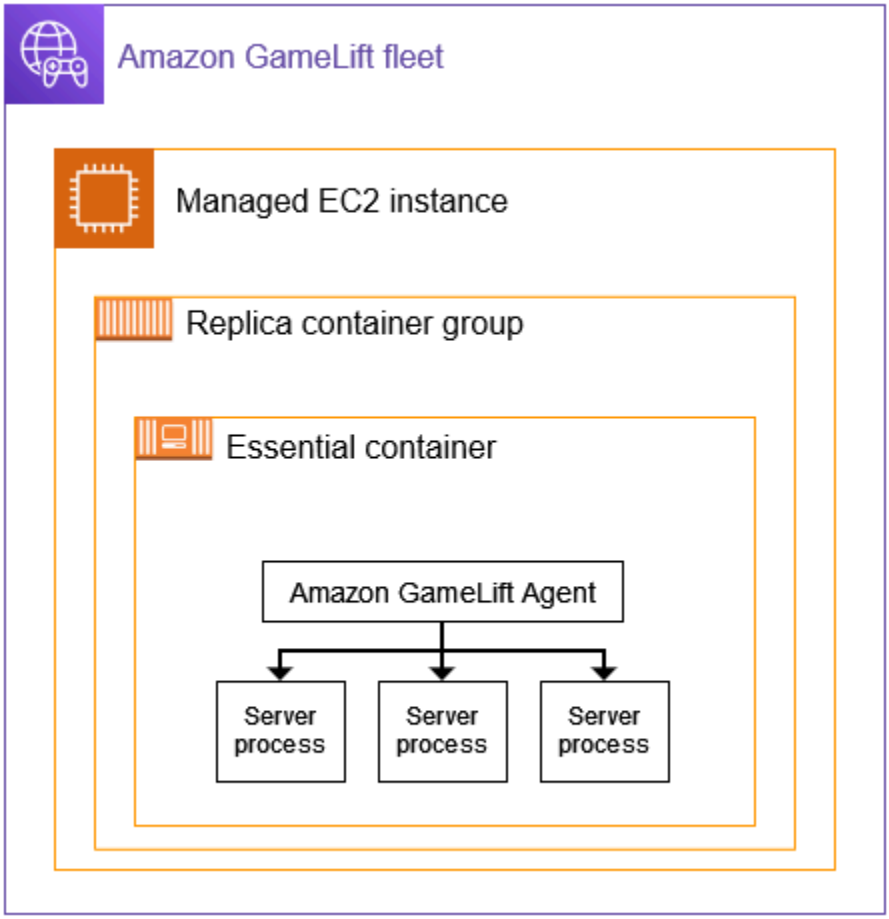
- Les conteneurs Daemon exécutent un service daemon pour gérer les processus en arrière-plan. Une utilisation courante d'un conteneur daemon consiste à exécuter un agent Amazon CloudWatch (CloudWatch) pour collecter des métriques, des journaux et des traces pour vos conteneurs. Les conteneurs daemon peuvent être essentiels ou non essentiels selon le moment où une panne de conteneur doit entraîner le redémarrage d'un groupe de conteneurs.

Calcul

Un ordinateur est une ressource d'hébergement de flotte enregistrée auprès du GameLift service Amazon et capable de communiquer avec le service. Dans un parc de conteneurs, un ordinateur est un conteneur doté d'un processus qui gère le processus d'enregistrement des ordinateurs. Dans le conteneur réplique essentiel d'une flotte de conteneurs, l' GameLift agent Amazon enregistre automatiquement ce conteneur en tant que conteneur informatique.

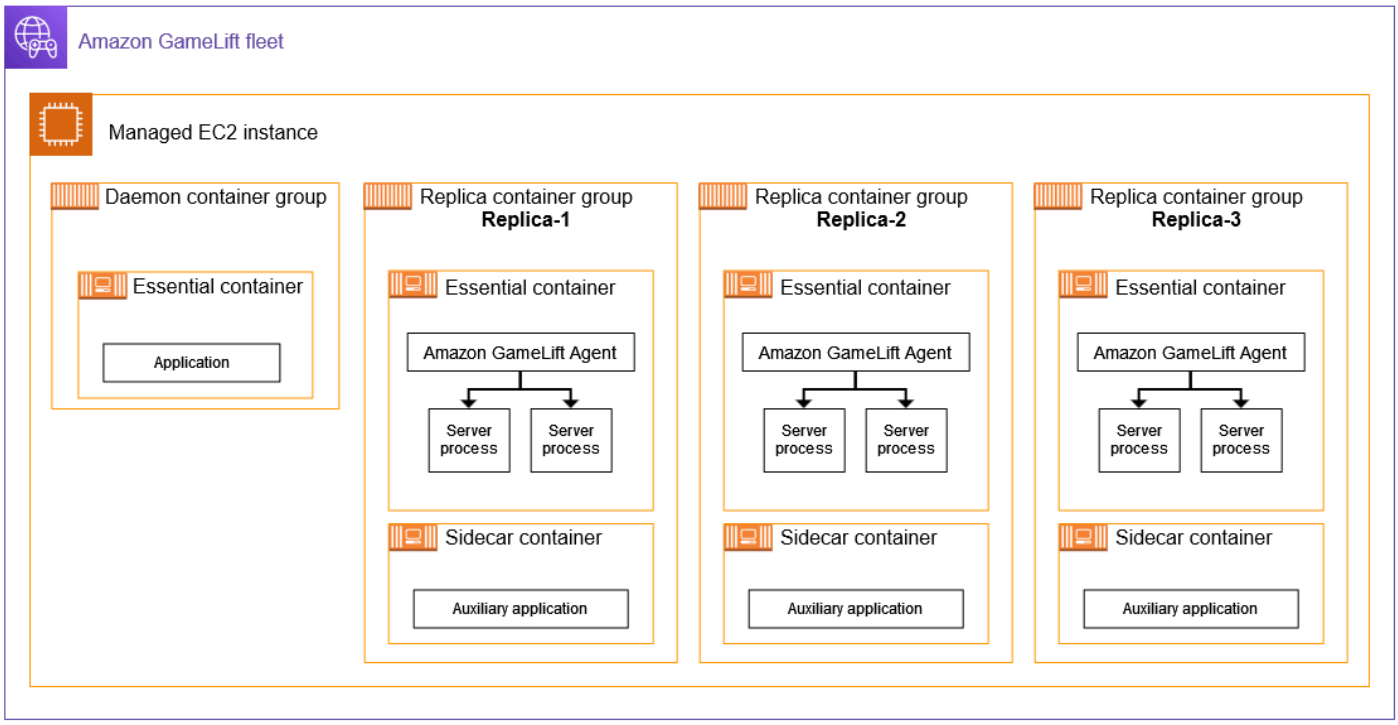
Architectures communes

Le schéma suivant illustre la structure de flotte de conteneurs la plus simple. Dans cette structure, chaque instance du parc conserve une copie du groupe de conteneurs de répliques. Le groupe de conteneurs possède un seul conteneur essentiel qui exécute l' GameLift agent Amazon, l'application du serveur de jeu et tous les logiciels compatibles pour l'hébergement de sessions de jeu. L'agent met en œuvre des instructions spécifiques à la flotte pour exécuter trois processus de serveur simultanément. Comme il existe un groupe de conteneurs de répliques par instance, chaque instance de flotte exécute trois processus de serveur simultanément.



Ce deuxième exemple illustre une conception de flotte de conteneurs plus complexe. Dans cet exemple, la flotte possède un groupe de conteneurs répliqué avec plusieurs conteneurs et un groupe de conteneurs daemon avec un conteneur. La configuration de flotte place trois copies du groupe de conteneurs de répliques sur chaque instance de flotte. Le groupe de conteneurs daemon n'est jamais répliqué.

Chaque ensemble de conteneurs de groupes de répliques du contient trois copies sur chaque instance. Dans chaque conteneur de répliques essentiel, l'agent est chargé d'exécuter deux processus serveur simultanément. Par conséquent, chaque instance de flotte exécute six processus de serveur simultanément (deux processus dans chacun des trois conteneurs de répliques essentiels).



Concepts de base

Cette section résume la manière dont Amazon GameLift met en œuvre certains concepts de base relatifs aux conteneurs. Pour obtenir des instructions sur la manière de travailler avec des flottes de conteneurs, consultez les rubriques pertinentes de ce guide.

Emballage de groupes de conteneurs

Lorsque vous développez votre structure de conteneurs en vue d'un déploiement dans une flotte de conteneurs, un objectif commun est d'optimiser l'utilisation de la puissance de calcul disponible. Pour atteindre cet objectif, trouvez le plus grand nombre de répliques de groupes de conteneurs que vous pouvez placer sur une instance de flotte sans affecter les performances du serveur de jeu.

Amazon GameLift peut vous y aider. Il calcule le nombre maximum de groupes de répliques par instance, sur la base des informations suivantes :

- Le type d'instance du parc et les ressources de processeur et de mémoire disponibles.
- Les exigences en termes de processeur et de mémoire que vous définissez pour tous les conteneurs de votre groupe de répliques.

  Les exigences en termes de processeur et de mémoire que vous définissez pour tous les conteneurs d'un groupe de démons, le cas échéant.

- Ressources réservées à la gestion des conteneurs et autres applications critiques sur chaque instance.

Lorsque vous créez une flotte de conteneurs, vous pouvez choisir d'utiliser le nombre maximum calculé ou de remplacer le nombre calculé en spécifiant le nombre souhaité. La meilleure pratique consiste à tester le logiciel de votre serveur de jeu conteneurisé afin de déterminer avec précision les besoins en ressources. Utilisez ces données pour trouver une stratégie d'emballage optimale pour les performances du serveur de jeu.

Les serveurs de jeu et l' GameLiftagent Amazon

Lorsque vous créez votre réplica de conteneur essentiel, vous regroupez le logiciel de votre serveur de jeu et l' GameLift agent Amazon dans la même image de conteneur. Cet agent informatique contrôle le cycle de vie des serveurs de jeu dans le conteneur. Dans chaque groupe de conteneurs de répliques, le conteneur de répliques essentiel exécute l'agent et tous les processus du serveur de jeu.

L' GameLift agent Amazon exécute les instructions dans la configuration d'exécution de la flotte de conteneurs. La configuration d'exécution identifie (1) l'exécutable à exécuter, (2) un ensemble facultatif de paramètres de lancement et (3) le nombre de processus à exécuter simultanément. Une configuration d'exécution peut contenir des instructions pour plusieurs exécutables différents. Au moins une instruction doit être exécutable pour le serveur de jeu. Par exemple, une configuration d'exécution peut demander à l'agent de gérer 10 processus du fichier exécutable de votre serveur de jeu pour une utilisation en production, un processus du même exécutable avec des paramètres de lancement spéciaux à des fins de test et un processus pour un utilitaire de journalisation.

Vous pouvez modifier la configuration d'exécution d'une flotte à tout moment. L' GameLift agent Amazon demande régulièrement des mises à jour au service. Lorsqu'une configuration d'exécution mise à jour est disponible, l'agent la reçoit et commence à mettre en œuvre les instructions. Les actions peuvent inclure l'ajout ou l'arrêt de processus du serveur.

L' GameLift agent Amazon est une version open source de l'agent informatique qu'Amazon GameLift utilise pour les flottes EC2 gérées. Ce guide fournit des instructions sur la façon de créer l'agent à partir de la source et de l'intégrer dans une image de conteneur. L'agent exécute les tâches suivantes :

Gestion des processus du serveur :

- Démarrez, arrêtez et remplacez les processus du serveur en fonction de la configuration d'exécution.
- Arrêtez les processus du serveur lorsqu'ils ne s'activent pas à temps.
- Signalez à Amazon GameLift la fin d'un processus serveur.
- Émettez des événements de flotte pour les processus du serveur.

Gestion des conteneurs :

- Arrêtez les processus du serveur en réponse aux demandes d'Amazon GameLift.
- Signalez l'état du conteneur.

Tâches de téléchargement du journal :

- Téléchargez les journaux de session de jeu dans un compartiment Amazon S3 désigné.
- Téléchargez les journaux des agents informatiques dans un compartiment Amazon S3 désigné.

Dimensionnement de la capacité d'une flotte

La capacité de la flotte mesure le nombre de sessions de jeu que la flotte peut héberger à tout moment. Vous pouvez également mesurer la capacité en fonction du nombre de joueurs que la flotte peut supporter simultanément.

Pour augmenter ou diminuer la capacité d'hébergement d'une flotte, vous ajoutez ou supprimez des instances de flotte. La stratégie d'emballage d'une flotte de conteneurs détermine le nombre de sessions de jeu exécutées simultanément sur chaque instance de flotte. Ce nombre indique le nombre de sessions de jeu (et de machines à sous) que vous ajoutez ou retirez lorsque vous augmentez ou diminuez la capacité de la flotte.

Avec les flottes de conteneurs, vous pouvez utiliser n'importe laquelle des méthodes de dimensionnement proposées par Amazon GameLift. Il s'agit des licences suivantes :

- Définissez la capacité du parc manuellement en définissant le nombre d'instances de flotte souhaité.
- Configurez le dimensionnement automatique en ciblant la mémoire tampon souhaitée parmi les instances disponibles (suivi des cibles). Cette méthode maintient automatiquement un ensemble de ressources d'hébergement inactives afin que les joueurs entrants puissent toujours accéder

rapidement aux jeux. Au fur et à mesure que la demande des joueurs augmente ou diminue, la taille de cette zone tampon est continuellement ajustée.

- Configurez la mise à l'échelle automatique avec des règles de mise à l'échelle personnalisées (fonctionnalité avancée).

Connexions client/serveur de jeu

Les flottes EC2 et les flottes de conteneurs gérés gèrent les connexions entre les clients de jeu et les serveurs de jeux hébergés dans le cloud de la même manière. Lorsqu'Amazon GameLift crée une nouvelle session de jeu, le service communique les informations de connexion de la session de jeu. Les clients du jeu utilisent ces informations pour se connecter directement au serveur de jeu hébergeant la session de jeu. Pour tous les types de flottes, les informations de connexion se composent d'une adresse IP et d'une attribution de port.

Lorsque vous créez une flotte de conteneurs, vous définissez deux ensembles de plages de ports. Tout d'abord, vous définissez une série de ports de connexion orientés vers l'extérieur qui permettent aux clients du jeu de se connecter à un jeu. Ensuite, vous définissez un ensemble de ports de conteneur internes uniquement, qui sont affectés à chaque processus de serveur de jeu exécuté dans le conteneur. Amazon mappe GameLift dynamiquement les ports de conteneurs internes aux ports de connexion externes pour permettre aux joueurs d'accéder aux jeux. Cette approche fournit un niveau de sécurité supplémentaire en protégeant vos serveurs de jeu d'un accès direct aux ports de conteneurs.

Lorsque vous définissez des plages de ports pour un parc de conteneurs, vous devez fournir des plages comportant suffisamment de ports pour prendre en charge tous les processus serveur exécutés simultanément sur les conteneurs d'une instance.

Pour un contrôle supplémentaire, vous définissez également les autorisations entrantes pour une flotte. Les autorisations entrantes déterminent quels ports de connexion sont ouverts pour le trafic entrant. Vous pouvez modifier les autorisations d'entrée d'une flotte à tout moment. Avec les autorisations entrantes, vous pouvez rapidement fermer tous les ports de connexion, en ouvrir certains ou les ouvrir tous selon vos besoins.

Feuille de route de développement pour les GameLift conteneurs Amazon

Cette documentation concerne une fonctionnalité en version préliminaire publique. Elle est susceptible d'être modifiée.

Le flux de travail suivant résume les étapes à suivre pour faire fonctionner vos serveurs de jeu sur un parc de GameLift conteneurs Amazon.

Étape 1 : Intégrez votre jeu à Amazon GameLift

Ajoutez des fonctionnalités à votre serveur de jeu afin qu'il puisse communiquer avec le GameLift service Amazon lorsqu'il est déployé sur une flotte de conteneurs. Si vous utilisez le FlexMatch matchmaking, ajoutez cette fonctionnalité à votre serveur de jeu et à votre client. Pour plus d'informations, consultez Intégrez votre jeu à Amazon GameLift.

- Procurez-vous le SDK GameLift du serveur Amazon (version 5+) et configurez-le avec votre projet de jeu. Le SDK du serveur est disponible en C++, C# et Go.
- Modifiez le code de votre serveur de jeu pour ajouter les fonctionnalités requises du SDK du serveur.
- Package de la version de votre serveur de jeu pour Linux. Si vous développez sous Windows, cette étape peut nécessiter un travail supplémentaire pour configurer un environnement Linux.
- (Facultatif) Testez l'intégration de votre serveur de jeu à l'aide d'une GameLift Anywhere flotte Amazon. Effectuez un test avant de préparer votre image de conteneur afin d'isoler les problèmes liés à votre travail d'intégration. Pour tester les connexions client/serveur de jeu, intégrez également votre client de jeu.

> **Note**
>
> Si vous développez sous Windows, configurez un espace de travail Linux distinct ou utilisez un outil tel que le sous-système Windows pour Linux (WSL). Vous aurez besoin d'un environnement Linux pour tester la version de votre serveur de jeu, ainsi que pour créer et tester vos images de conteneur.

Étape 2 : Préparez l'image du conteneur de votre serveur de jeu

Créez une image de conteneur qui exécute les processus de votre serveur de jeu et stockez-la dans un référentiel Amazon Elastic Container Registry (Amazon ECR) à utiliser avec Amazon.

GameLift Pour obtenir des instructions complètes, veuillez consulter Préparez une image de conteneur avec le logiciel de votre serveur de jeu.

- Configurez un répertoire de travail pour votre image de conteneur, avec le build de votre jeu Linux, le script d'installation, ainsi que tous les logiciels et dépendances associés.
- Obtenez le code source d'Amazon GameLift Agent, créez-le et ajoutez le `jar` fichier à votre répertoire de travail.
- Obtenez le Dockerfile par défaut et modifiez-le pour configurer une image de conteneur avec le logiciel de votre serveur de jeu.
- Créez l'image de votre conteneur. Effectuez cette étape dans un environnement Linux.
- Créez un référentiel privé Amazon ECR et envoyez-y l'image de votre conteneur. Créez le dépôt dans celui-ci Compte AWS et à l' Région AWS endroit où vous prévoyez de déployer votre flotte de conteneurs.
- (facultatif) Testez les images de vos conteneurs à l'aide de votre Anywhere flotte. Vous pouvez définir une configuration d'exécution pour transmettre des instructions à l' GameLift agent Amazon.

Étape 3 : Créez vos conteneurs et groupes de conteneurs

Concevez une architecture de conteneur pour l'hébergement de jeux sur Amazon GameLift. Consultez Concevez une flotte de GameLift conteneurs Amazon et Création de définitions de groupes de conteneurs pour une flotte de GameLift conteneurs Amazon.

- Définissez les configurations de vos conteneurs. Pour chaque conteneur, vous allez définir des problèmes tels que les processus d'exécution, l'allocation de mémoire, les contrôles de santé, les ports réseau, etc.
- Utilisez la GameLift console Amazon ou la AWS CLI pour créer des définitions de groupes de conteneurs avec vos configurations de conteneurs. Lorsque vous créez une définition de groupe de conteneurs, Amazon GameLift prend un instantané de chaque image de conteneur à ce moment-là.

Étape 4 : Déployez votre serveur de jeu conteneurisé sur une flotte de conteneurs

Utilisez les définitions de groupes de conteneurs créées à l'étape précédente pour créer une flotte de conteneurs et déployer votre logiciel de serveur de jeu conteneurisé. veuillez consulter Création d'une flotte de GameLift conteneurs Amazon.

- Utilisez la GameLift console Amazon ou la AWS CLI pour créer un parc de conteneurs.

- Suivez l'état de la flotte à mesure que les instances de flotte sont déployées et activées. Vérifiez les événements de création de flotte pour vérifier que la flotte est correctement déployée sur tous les sites.
- Vérifiez que les clients du jeu peuvent demander et rejoindre des sessions de jeu et jouer au jeu. Si vous avez configuré le matchmaking, testez ces scénarios.

Étape 5 : Gérez vos flottes

Alors que vous vous préparez à une utilisation en production, élaborez votre solution d'hébergement de jeux et gérez le cycle de vie de votre hébergement.

- Créez des flottes multi-sites et des flottes dans d'autres sites Régions AWS pour soutenir votre base de joueurs.
- Configurez le placement de l'hébergement du jeu avec des files d'attente ou le FlexMatch matchmaking. Consultez ces ressources :
  - Configuration des GameLift files d'attente Amazon pour le placement des sessions de jeu
  - FlexMatch Manuel du développeur
- Configurez le dimensionnement automatique pour gérer la capacité de la flotte en fonction de la demande des joueurs pour les sessions de jeu.
- Configurez la surveillance de vos flottes de conteneurs. Utilisez GameLift les métriques d'Amazon, récupérez les journaux de sessions de jeu et les journaux de conteneurs, configurez l'accès à distance à des conteneurs individuels.
- Mettre en place une gestion à long terme des flottes de conteneurs. Utilisez des alias de flotte pour rationaliser le processus de mise à jour des flottes de conteneurs. Créez des AWS CloudFormation modèles pour gérer le cycle de vie de votre flotte. Consultez ces ressources :
  - Ajouter un alias à une GameLift flotte Amazon
  - Gérez les ressources à l'aide AWS CloudFormation

Intégrez votre jeu à Amazon GameLift

Cette documentation concerne une fonctionnalité en version préliminaire publique. Elle est susceptible d’être modifiée.

Avant de créer une image de conteneur avec le logiciel de votre serveur de jeu et de la déployer sur Amazon GameLift pour l'hébergement dans le cloud, intégrez d'abord votre projet de jeu au SDK GameLift du serveur Amazon et créez un serveur de jeu compatible avec Linux. Cette rubrique présente les différents outils d'intégration proposés par Amazon GameLift.

Les serveurs de jeux hébergés doivent être en mesure de communiquer avec le GameLift service Amazon. Configurez la communication en ajoutant le SDK GameLift du serveur Amazon (version 5+) à votre projet de jeu et en modifiant le code du serveur de votre jeu. Amazon GameLift fournit des ressources et de la documentation relatives au SDK pour serveurs compatibles avec plusieurs langages et moteurs de jeu.

Le processus d'intégration des serveurs de jeux conteneurisés est pratiquement identique à celui des serveurs de jeux destinés à être hébergés sur des flottes gérées par EC2 ou Amazon. GameLift Anywhere

Outils d'intégration

Amazon GameLift fournit les outils et le support linguistique suivants pour l'intégration :

Pour les développeurs d'Unreal Engine

Utilisez le plugin léger pour Unreal. Ce plugin inclut les bibliothèques du SDK du serveur C++ avec les GameLift fonctionnalités Amazon requises. Utilisez la documentation pour configurer votre projet de jeu Unreal pour le plugin et mettez à jour le code de votre jeu avec les blocs de code fournis afin d'ajouter les fonctionnalités requises pour les versions de votre serveur et de votre client.

- Téléchargement du plugin SDK
- Guide : Intégrez votre projet Unreal à Amazon GameLift
- Guide de référence : C++ Server SDK 5 pour Unreal

Remarque : le plugin GameLift autonome Amazon pour Unreal Engine ne prend pas en charge l'utilisation de flottes de conteneurs.

Pour les développeurs Unity

Utilisez le plugin léger pour Unity. Ce plugin inclut les bibliothèques du SDK du serveur C# avec les fonctionnalités Amazon GameLift requises. Utilisez la documentation pour configurer votre projet de jeu Unreal pour le plugin et mettez à jour le code de votre jeu avec les blocs de code fournis afin d'ajouter les fonctionnalités requises pour les versions de votre serveur et de votre client.

- Téléchargement du plugin SDK
- Guide : Intégrez votre projet Unity à Amazon GameLift
- Guide de référence : SDK 5 pour serveur C# pour Unity

Remarque : le plugin GameLift autonome Amazon pour Unity ne prend pas en charge l'utilisation de flottes de conteneurs.

Pour les développeurs utilisant d'autres moteurs de jeu

Suivez ces instructions générales d'intégration du serveur et du client :

- Intégrer un serveur de jeu
- Intégrer un client de jeu

Amazon GameLift propose des bibliothèques du SDK 5 pour les serveurs dans les langues suivantes :

- Server SDK 5 pour C++ [Téléchargement du SDK] [Guide de référence]
- Server SDK 5 pour C# [Téléchargement du SDK] [Guide de référence]
- Server SDK 5 pour Go [Téléchargement du SDK] [Guide de référence]

Concevez votre serveur de jeu pour Linux

Les flottes de GameLift conteneurs Amazon prennent en charge les serveurs de jeux qui s'exécutent sur une plateforme Linux. Voici quelques conseils pour créer votre serveur de jeu pour une cible Linux :

- Si vous développez votre jeu avec le moteur de jeu Unity, l'éditeur de jeu fournit un support intégré sans aucune exigence particulière pour le compiler pour Linux.
- Si vous développez votre jeu en C++, vous devez inclure les bibliothèques OpenSSL pour Linux lorsque vous créez le SDK du serveur GameLift Amazon pour C++ et lorsque vous créez votre serveur de jeu. Incluez également les mêmes bibliothèques dans l'image du conteneur de votre serveur de jeu.
- Si vous développez votre jeu avec Unreal Engine sous Windows, envisagez les options suivantes :
  - Travaillez avec Unreal Engine pour configurer une chaîne d'outils de compilation croisée.
  - Configurez un espace de travail Linux distinct ou utilisez un outil tel que le sous-système Windows pour Linux (WSL). Vous pouvez utiliser cet environnement pour exécuter l'éditeur Unreal sous Linux afin de créer votre serveur de jeu.

Testez votre intégration localement

Vous pouvez tester l'intégration de votre jeu localement à l'aide d'une GameLift Anywhere flotte Amazon. Cette approche est une bonne pratique pour aider à isoler les problèmes directement liés à l'intégration. Une Anywhere flotte est un outil utile pour exécuter des applications de test et des scénarios de jeu, tels que le démarrage/l'arrêt de sessions de jeu et le suivi des connexions entre joueurs. Vous pouvez créer et tester de manière itérative beaucoup plus rapidement avec une Anywhere flotte, ce qui offre une meilleure visibilité sur l'activité d'hébergement.

Consultez Testez votre intégration à l'aide des GameLift Anywhere flottes Amazon pour obtenir de l'aide sur l'utilisation d'une GameLift Anywhere flotte Amazon pour les tests d'intégration. Le processus de configuration d'un environnement de test se présente comme suit :

1. Configurez un appareil local qui exécute Linux.
2. Configurez une Anywhere flotte. Créez un emplacement personnalisé pour votre appareil local, créez un Anywhere parc, puis enregistrez votre appareil local comme ordinateur dans le parc.
3. Obtenez un jeton d'authentification pour votre serveur de jeu. Votre processus de serveur intégré nécessite un jeton pour vous authentifier auprès du GameLift service Amazon. Vous pouvez réutiliser le même jeton pour plusieurs processus de serveur exécutés simultanément. Cette étape n'est requise que lors de l'utilisation d'un Anywhere parc pour les tests d'intégration.

   > **Note**
   >
   > Les jetons d'authentification sont temporaires et doivent être régulièrement actualisés. Envisagez d'ajouter un script au package de compilation de votre serveur pour demander un nouveau jeton.

4. Mettez à jour le code de votre serveur de jeu pourAnywhere. Lors de l'exécution sur une flotte Anywhere, le serveur de jeu doit appeler l'action du SDK du serveur `InitSdk()` (C++) (C#) (Unreal) avec les paramètres de serveur suivants. Cette étape n'est requise que lors de l'utilisation d'une Anywhere flotte pour les tests d'intégration. Une fois que vous avez ajouté l' GameLift agent Amazon à l'image de votre conteneur, il gère automatiquement ces paramètres.

   Il est recommandé de configurer le code de votre serveur pour extraire ces valeurs des variables d'environnement ou des arguments de console que vous spécifiez au lancement.

   - `webSocketUrl`— Utilise la valeur de`GameLiftServiceSdkEndpoint`, qui est renvoyée par l'appel à`register-compute`.

- `processId`— Attribuez un identifiant unique au processus du serveur.
- `fleetId`— L'identifiant de flotte Anywhere, qui est renvoyé lors de l'appel à`create-fleet`.
- `authToken`— Un jeton d'authentification valide, renvoyé par l'appel à`get-compute-auth-token`.

5. Sur votre machine locale, configurez le logiciel de compilation de votre serveur de jeu et lancez un processus serveur.

   Si l'intégration de votre serveur est réussie, le processus du serveur appelle l'action du SDK du serveur `InitSDK()` pour établir la connexion avec le GameLift service Amazon, suivie d'un appel `ProcessReady()` pour informer le service qu'il est prêt à héberger une session de jeu.

6. Démarrez une session de jeu. Si vous avez intégré votre client de jeu pour demander une session de jeu, vous pouvez l'utiliser pour demander une nouvelle session de jeu. Si ce n'est pas le cas, utilisez la commande AWS CLI create-game-session. Amazon GameLift crée un `GameSession` objet et lance le processus pour démarrer une nouvelle session de jeu.

   Si votre intégration fonctionne, Amazon GameLift appelle un processus serveur sur votre poste de travail local pour démarrer une nouvelle session de jeu (en utilisant le `onStartGameSession()` rappel). Lorsqu'une session de jeu est prête pour les joueurs, le processus du serveur appelle`ActivateGameSession()`. En réponse, Amazon GameLift met à jour le `GameSession` statut et les informations de connexion afin qu'un client de jeu puisse se connecter à la session de jeu et jouer au jeu.

Préparez une image de conteneur avec le logiciel de votre serveur de jeu

Cette documentation concerne une fonctionnalité en version préliminaire publique. Elle est susceptible d’être modifiée.

Le conteneur est l'élément le plus fondamental d'une flotte de GameLift conteneurs Amazon. Votre conteneur inclut votre serveur de jeu, ainsi que ses dépendances telles que les SDK, les logiciels, les répertoires et les fichiers.

Pour fonctionner dans une flotte de conteneurs, votre serveur de jeu doit fonctionner sous Linux et être intégré au SDK 5.x du serveur.

Rubriques

- Configurez votre répertoire de travail
- Créez l'image de votre conteneur
- Transférez l'image de votre conteneur vers Amazon ECR

Configurez votre répertoire de travail

Votre répertoire de travail est l'endroit où vous placez tous les fichiers dont vous avez besoin pour créer votre image de conteneur et définir la manière dont Amazon l' GameLift exécute.

Pour configurer le répertoire de travail de votre conteneur

1. Créez le répertoire dans lequel vous souhaitez travailler avec les images de vos GameLift conteneurs Amazon.

   Example

   Par exemple :

   ```
   [~/]$ mkdir -p work/glc/gamebuild && cd work && find .
   .
   ./glc
   ./glc/gamebuild
   ```

2. Clonez l' GameLift agent Amazon.

   Example

   Par exemple :

   ```
   [~/work]$ git clone https://github.com/aws/amazon-gamelift-agent.git
   Cloning into 'amazon-gamelift-agent'...
   ```

3. Construisez-le à l' GameLiftAgent aide de Maven.

   Example

   Par exemple :

   ```
   [~/work]$ cd amazon-gamelift-agent
   ```

Example

```
[~/work/amazon-gamelift-agent]$ mvn clean compile assembly:single && \
mv target ../glc && cd .. && find glc
```

4. Ajoutez un serveur de jeu intégré au SDK 5.x du serveur, intégré et intégré dans un `.ZIP` fichier.
5. Copiez votre `.ZIP` fichier dans`~/work/glc/gamebuild/`.

Si vous n'avez pas de serveur de jeu SDK 5.x, vous pouvez télécharger et utiliser notre exemple de SimpleServerjeu pour essayer d'utiliser une flotte de conteneurs.

Example

```
[~/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' &&
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
Dload  Upload   Total   Spent    Left  Speed
100 5140k  100 5140k    0     0  12.3M      0 --:--:-- --:--:-- --:--:-- 12.3M
glc
glc/target
glc/target/GameLiftAgent-1.0.jar
glc/gamebuild
glc/gamebuild/SimpleServer.zip
```

Créez l'image de votre conteneur

Votre Dockerfile spécifie l'environnement, le logiciel et les instructions pour créer votre conteneur.

Pour créer votre Dockerfile

1. Accédez au `glc` sous-répertoire.

   Example

   ```
   [~/work]$ cd glc && find

   .
   ./target
   ```

```
./target/GameLiftAgent-1.0.jar
./gamebuild
```

2. Créez et ouvrez un nouveau Dockerfile.

   Example

   Par exemple :

   ```
   [~/work/glc]$ nano Dockerfile
   ```

3. Copiez à partir de l'un des modèles suivants, puis collez le contenu dans votre Dockerfile.

Modèle Dockerfile pour votre serveur de jeu

Ce modèle contient les instructions minimales dont un conteneur a besoin pour être utilisable dans une GameLift flotte Amazon. Modifiez le contenu selon les besoins de votre serveur de jeu.

```
# 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
```

Dockerfile pour l'exemple **`SimpleServer`**

```
# 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="SimpleServer.zip" \
  #
# 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="GameLiftSampleServer" \
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 ./target/$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
```

> **Note**
>
> Remarque : Certaines variables d'environnement du Dockerfile peuvent être remplacées par le. ContainerDefinition

Pour créer l'image de votre conteneur

1. Créez l'image de votre conteneur.

   Si vous utilisez votre propre serveur SDK 5.x

   Vous pouvez spécifier le nom du dépôt local de votre choix.

   Example

   ```
   [~/work/glc]$ docker build -t <local repository name>:<optional tag> .
   ```

   Si vous utilisez notre **`SimpleServer`** échantillon

   Example

   ```
   [~/work/glc]$ docker build -t simple-server:version-1 .
   Successfully built 0123456789012
   Successfully tagged simple-server:version-1
   ```

   > **Note**
   >
   > Dans les exemples suivants, nous utilisons *`simpler-server`* comme `REPOSITORY` valeur initiale et `version-1` comme valeur. `TAG`

2. Consultez la liste des images et notez les `IMAGE ID` valeurs `REPOSITORY` et. Vous en aurez besoin dans le cadre de la procédure ci-dessous.

Example

```
[~/work/glc]$ docker images
REPOSITORY       TAG          IMAGE ID        CREATED           SIZE
simple-server    version-1    0123456789012    14 minutes ago    1.24GB
```

Transférez l'image de votre conteneur vers Amazon ECR

Téléchargez l'image de votre conteneur dans un référentiel privé sur Amazon ECR. Lorsque vous créez une définition de groupe de conteneurs, vous faites référence à cet emplacement de référentiel afin qu'Amazon GameLift puisse prendre un instantané de l'image de votre conteneur et l'utiliser lors du déploiement d'un parc de conteneurs.

> **Note**
>
> Si vous ne possédez pas encore de référentiel privé Amazon ECR, créez-en un.

Pour obtenir vos informations d'identification Amazon ECR

- Avant de pouvoir transférer l'image de votre conteneur vers Amazon ECR, vous devez obtenir vos AWS informations d'identification sous forme temporaire et les fournir à Docker. Obtenez vos informations d'identification Amazon ECR pour que Docker puisse se connecter.

  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
  ```

Pour transférer l'image de votre conteneur vers Amazon ECR

1. Copiez l'URI du référentiel privé Amazon ECR que vous souhaitez utiliser.

2. Appliquez une balise Amazon ECR à l'image de votre conteneur.

   Example

   ```
   [~/work/glc]$ docker tag <IMAGE ID from above> <Amazon ECR private repository
    URI>:<optional tag>
   ```

3. Transférez l'image de votre conteneur vers Amazon ECR

   Example

   ```
   [~/work/glc]$ docker image push  <Amazon ECR private repository URI>
   ```

Concevez une flotte de GameLift conteneurs Amazon

Cette documentation concerne une fonctionnalité en version préliminaire publique. Elle est susceptible d'être modifiée.

Ces rubriques présentent les principales décisions que vous prendrez lors de la configuration d'une flotte de GameLift conteneurs Amazon. Vos décisions ont un impact sur la façon dont vous configurez les paramètres des conteneurs, des groupes de conteneurs et des flottes.

Rubriques

- Architectez la structure de votre flotte de conteneurs
- Définissez des limites de ressources
- Désignez les contenants essentiels
- Configuration des connexions réseau
- Configurez des contrôles de santé pour les conteneurs
- Définir les dépendances des conteneurs
- Configuration d'une flotte de conteneurs

Architectez la structure de votre flotte de conteneurs

Dans un premier temps, identifiez les logiciels et les ressources nécessaires pour héberger votre serveur de jeu, notamment les suivants :

- Votre application de serveur de jeu. L'application doit être intégrée aux GameLift fonctionnalités d'Amazon pour l'hébergement, y compris le SDK du serveur version 5+. veuillez consulter Intégrez votre jeu à Amazon GameLift.
- L' GameLift agent Amazon. Cet agent informatique assure la communication avec le GameLift service Amazon et gère le cycle de vie de tous les processus du serveur de jeu. Pour en savoir plus, consultez Les serveurs de jeu et l' GameLiftagent Amazon.
- Logiciels et ressources supplémentaires selon les besoins. Cela peut inclure les logiciels nécessaires pour exécuter les applications de votre serveur de jeu. Les logiciels de support courants sont utilisés pour la journalisation et la surveillance, la sécurité, la diffusion de contenu et la synchronisation des données.

Ensuite, déterminez comment structurer votre logiciel et vos ressources pour une flotte de GameLift conteneurs Amazon. Amazon GameLift utilise des groupes de conteneurs pour organiser les conteneurs. Une flotte possède toujours un groupe de conteneurs répliques et peut éventuellement comporter une flotte de conteneurs daemon. Pour en savoir plus, consultez Composants du parc de conteneurs.

- Commencez par concevoir votre groupe de répliques de conteneurs. Considérez les directives suivantes :
  - Regroupez votre application de serveur de jeu et l' GameLift agent Amazon dans le même conteneur. Faites de ce conteneur le seul conteneur essentiel du groupe de répliques.
  - Organisez tous les autres logiciels de votre serveur de jeu dans des conteneurs. Vous pouvez choisir de tout placer dans un seul conteneur du groupe de répliques. Vous pouvez également choisir de créer un ou plusieurs conteneurs de sidecar. Voici quelques raisons d'utiliser les sidecars :
    - Pour configurer une séquence de démarrage/arrêt pour un logiciel individuel. Vous pouvez y parvenir en plaçant les logiciels dans des conteneurs distincts et en configurant des dépendances entre eux.
    - Pour définir des limites d'utilisation de la mémoire et du processeur spécifiques au conteneur.
    - Pour spécifier différents paramètres de configuration de conteneur pour chaque conteneur, tels qu'une commande de lancement, un point d'entrée, un répertoire de travail, des variables d'environnement ou des contrôles de santé.
- Décidez si vous avez besoin d'un groupe de conteneurs daemon pour votre flotte. Éléments à prendre en compte :

- Les conteneurs Daemon sont généralement utilisés pour exécuter des processus d'arrière-plan ou de surveillance.
- Les conteneurs d'un groupe de démons ne sont pas répliqués sur une instance de flotte. Cela signifie que les conteneurs d'un groupe de démons ne s'adaptent pas au même rythme que le groupe de conteneurs de répliques.
- Un groupe de démons peut avoir plusieurs conteneurs. Vous pouvez désigner n'importe quel conteneur d'un groupe de démons comme essentiel.

Définissez des limites de ressources

Pour chaque groupe de conteneurs, déterminez la quantité de mémoire et de processeur dont le groupe a besoin pour exécuter son logiciel. Amazon GameLift s'appuie sur ces informations pour gérer les ressources du groupe de conteneurs. Il utilise également ces informations pour calculer le nombre de répliques de groupes de conteneurs qu'une image de flotte peut contenir. Vous pouvez également définir des limites pour des conteneurs individuels.

Définissez des limites facultatives pour les conteneurs

La définition de limites de ressources spécifiques aux conteneurs vous permet de mieux contrôler la manière dont les conteneurs individuels peuvent utiliser les ressources du groupe. Si vous ne définissez pas de limites spécifiques aux conteneurs, tous les conteneurs du groupe partagent les ressources du groupe. Le partage offre une plus grande flexibilité pour utiliser les ressources là où elles sont nécessaires. Cela augmente également le risque de concurrence entre les processus et d'entraîner une défaillance des conteneurs.

Définissez l'une des `ContainerDefinition` propriétés suivantes pour n'importe quel conteneur.

- `SoftLimit`(mémoire) — Réservez une quantité minimale de mémoire pour l'usage exclusif du conteneur. Le conteneur dispose toujours de la quantité réservée. Il peut dépasser ce minimum à tout moment, si des ressources supplémentaires sont disponibles.
- `HardLimit`(mémoire) — Définissez une limite de mémoire maximale pour le conteneur. Si le conteneur dépasse cette limite, cela entraîne un redémarrage.
- `Cpu`limite — Réservez un minimum de ressources CPU à l'usage exclusif du conteneur. Le conteneur dispose toujours de la quantité réservée. Il peut dépasser ce minimum à tout moment, si des ressources supplémentaires sont disponibles. (1024 unités de processeur équivalent à 1 vCPU.)

Définir des limites de ressources totales pour un groupe de conteneurs

Indiquez à Amazon GameLift la quantité de mémoire et de ressources CPU dont chaque groupe de conteneurs a besoin. L'objectif est d'allouer suffisamment de ressources pour optimiser les performances du serveur de jeu. Amazon GameLift utilise ces limites pour calculer comment regrouper des groupes de conteneurs de répliques sur une instance de flotte. Vous les utiliserez également lorsque vous choisirez un type d'instance pour un parc de conteneurs.

Calculez la mémoire totale et le processeur nécessaires pour tous les processus de chaque conteneur d'un groupe. Éléments à prendre en compte :

- Quels processus s'exécutent dans tous les conteneurs du groupe de conteneurs ? Additionnez les ressources nécessaires à ces processus.
- Combien de processus de serveur de jeu simultanés prévoyez-vous d'exécuter dans chaque groupe de conteneurs ? Vous définissez cette valeur dans le cadre de la configuration d'exécution d'une flotte, mais vous devez prévoir suffisamment de mémoire pour cette flotte ici (voir Optimisation de la configuration d'exécution).

Sur la base de votre estimation des exigences relatives aux groupes de conteneurs, définissez les `ContainerGroupDefinition` propriétés suivantes :

- `TotalMemoryLimit`— Définissez une limite de mémoire maximale pour le groupe de conteneurs. Tous les conteneurs du groupe partagent la mémoire allouée. Si vous définissez des limites de conteneur individuelles, la limite de mémoire totale doit être :
  - égal ou supérieur à la somme de toutes les limites de mémoire logicielle du conteneur
  - égal ou supérieur à la limite de mémoire dure la plus élevée pour un conteneur du groupe
- `TotalCpuLimit` — Définissez une limite maximale de CPU pour le groupe de conteneurs. Tous les conteneurs du groupe partagent les ressources CPU allouées. Si vous définissez des limites de conteneur individuelles, la limite totale du processeur doit être :
  - égal ou supérieur à la somme de toutes les limites du processeur du conteneur. Il est recommandé de définir cette valeur pour doubler la somme des limites du processeur du conteneur.

Exemple de scénario

Supposons que nous définissions un groupe de répliques contenant les trois conteneurs suivants :

- Le conteneur A est notre réplique de conteneur essentielle. Il exécute les processus du serveur de jeu et l' GameLift agent Amazon. Nous estimons les besoins en ressources d'un serveur de jeu à 512 MiB et 1024 CPU. Nous prévoyons de faire exécuter 10 processus serveur par

le conteneur. Comme ce conteneur exécute nos logiciels les plus critiques, nous avons défini une réserve de mémoire logicielle de 6 144 MiB et aucune limite de mémoire dure ni limite de réserve du processeur.

- Le conteneur B exécute des logiciels de support dont les besoins en ressources sont estimés à 1 024 Mo et 1 536 processeurs. Nous avons défini une limite de réserve de mémoire logicielle de 1 024 Mo, une limite de mémoire dure de 2 048 Mo et une limite de réserve du processeur de 1 024 Mo de processeur.
- Le conteneur C exécute des utilitaires de journalisation non critiques et d'autres utilitaires de surveillance. Nous avons défini une limite de mémoire dure de 512 Mo et une limite de réserve du processeur de 512 processeurs.

À l'aide de ces informations, nous avons défini les limites totales suivantes pour le groupe de conteneurs :

- Limite de mémoire totale : 7680 MiB. Cette valeur dépasse (1) la somme des limites de mémoire logicielle (6144+1024 Mo) et (2) la limite de mémoire dure la plus élevée (1024 Mo).
- Limite totale du processeur : 13312 processeurs. Cette valeur dépasse la somme de la limite du processeur (1024+512 processeurs).

Désignez les contenants essentiels

Pour chaque contenant, désignez le contenant comme étant essentiel ou non essentiel. Tous les groupes de conteneurs doivent comporter au moins un contenant essentiel. Le conteneur essentiel effectue le travail essentiel du groupe de conteneurs, comme l'hébergement de vos serveurs de jeu. On s'attend à ce que le contenant essentiel fonctionne toujours. En cas d'échec, l'ensemble du groupe de conteneurs redémarre.

- Le groupe de répliques de conteneurs de votre flotte peut contenir exactement un conteneur essentiel. Ce conteneur exécute l' GameLift agent Amazon et les processus du serveur de jeu qu'il gère.
- Si votre flotte possède un groupe de conteneurs daemon, vous pouvez désigner plusieurs conteneurs essentiels. Rendez un conteneur de démons essentiel si vous souhaitez qu'une défaillance de conteneur entraîne le redémarrage d'un groupe de conteneurs.

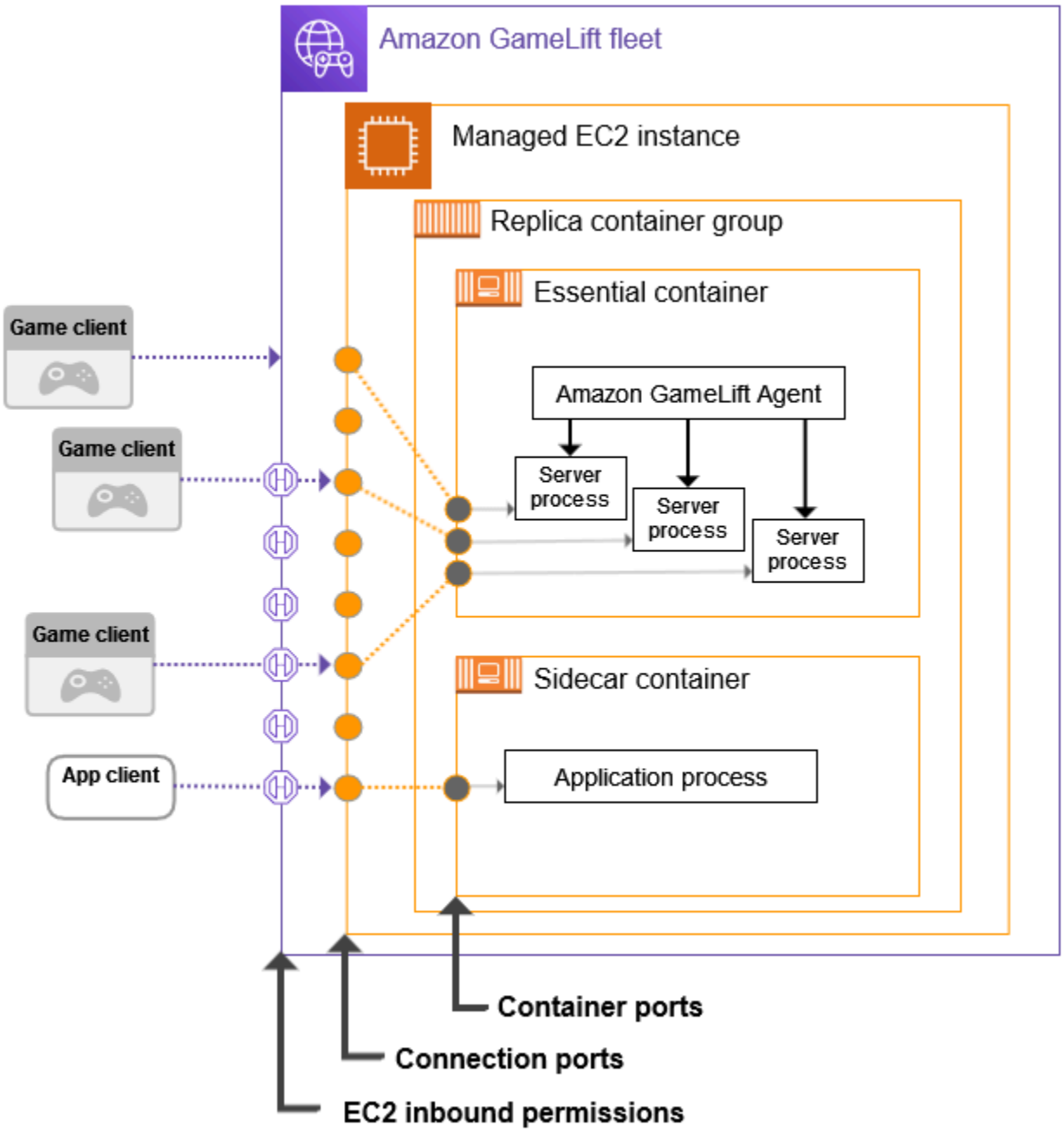
Définissez la `ContainerDefinition` propriété `Essential` sur true ou false pour chaque conteneur.

Configuration des connexions réseau

Vous pouvez établir un accès au réseau pour permettre au trafic externe de se connecter à n'importe quel conteneur d'une flotte de conteneurs. Par exemple, vous devez établir des connexions réseau avec le conteneur qui exécute les processus de votre serveur de jeu, afin que les clients du jeu puissent rejoindre votre jeu et y jouer. Les clients de jeu se connectent aux serveurs de jeux à l'aide de ports et d'adresses IP.

Dans une flotte de conteneurs, la connexion entre un client et un serveur n'est pas directe. En interne, un processus dans un conteneur écoute sur un port à conteneurs. En externe, le trafic entrant se connecte à une instance de flotte via un port de connexion. Amazon GameLift gère les mappages entre les ports de conteneurs internes et les ports de connexion externes, afin que le trafic entrant soit acheminé vers le processus approprié sur l'instance.

Amazon GameLift fournit un niveau de contrôle supplémentaire pour vos connexions réseau. Chaque flotte de conteneurs dispose d'un paramètre d'autorisations entrantes, qui vous permet de contrôler l'accès à chaque port de connexion externe. Vous ne pouvez pas modifier les configurations portuaires d'une flotte existante, mais vous pouvez autoriser ou restreindre l'accès selon les besoins en ajustant les autorisations entrantes. Par exemple, vous pouvez supprimer les autorisations pour tous les ports de connexion afin de bloquer tout accès aux conteneurs de la flotte.



Définissez les plages de ports pour conteneurs

Configurez une définition de conteneur avec suffisamment de ports de conteneur pour tout processus nécessitant un accès externe. Certains conteneurs n'auront pas besoin de ports. Les autres doivent disposer de suffisamment de ports pour en attribuer un à chaque processus qui en a besoin.

Votre groupe de répliques de conteneurs essentiel, qui gère vos serveurs de jeu, a besoin d'un port pour chaque processus de serveur de jeu exécuté simultanément (tel que configuré dans celui de la flotte`RuntimeConfiguration`). Le processus du serveur de jeu écoute le port attribué et le signale à Amazon GameLift.

Lorsque vous créez une définition de groupe de conteneurs, définissez une plage de ports de conteneurs pour chaque conteneur nécessitant un accès au réseau (voir ContainerDefinitionInput: PortConfiguration). Assurez-vous que la plage est suffisamment grande pour attribuer un port à chaque processus qui en a besoin. Les processus doivent se voir attribuer des numéros de port dans la configuration des ports du conteneur.

Définissez les plages de ports de connexion

Configurez votre flotte de conteneurs avec un ensemble de ports de connexion. Les ports de connexion fournissent un accès externe aux instances de flotte qui exécutent vos conteneurs. Amazon GameLift attribue des ports de connexion et les mappe aux ports de conteneurs selon les besoins.

Lorsque vous créez un parc de conteneurs, définissez une plage de ports de connexion (voir ContainerGroupsConfiguration: ConnectionPortRange). Assurez-vous que la plage possède suffisamment de ports pour être mappée à tous les ports de conteneurs d'une instance de flotte. Pour calculer le nombre minimal de ports de connexion nécessaires, utilisez la formule suivante :

```
[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]
```

Il est recommandé de doubler le nombre minimum de ports de connexion.

> **Note**
>
> Le nombre de ports de connexion peut potentiellement limiter le nombre de groupes de conteneurs de répliques par instance. Si une flotte ne dispose que de suffisamment de ports de connexion pour un seul groupe de conteneurs de répliques par instance, Amazon ne GameLift déploiera qu'un seul groupe de conteneurs de répliques, même si les instances disposent d'une puissance de calcul suffisante pour plusieurs groupes de conteneurs de répliques.

Définir les autorisations de trafic entrant

Les autorisations entrantes contrôlent l'accès externe à une flotte de conteneurs en spécifiant les ports de connexion à ouvrir pour le trafic entrant. Vous pouvez utiliser ce paramètre pour activer ou désactiver l'accès au réseau d'une flotte selon vos besoins.

Lorsque vous créez une flotte de conteneurs, définissez un ensemble d'autorisations entrantes (voir : CreateFleet EC2). InboundPermissions Définissez les propriétés du port d'autorisation entrant pour inclure certaines ou toutes les valeurs dans les paramètres des ports de connexion de la flotte. Pour modifier les autorisations entrantes sur une flotte de conteneurs existante, appelez UpdateFleetPortSettings.

Exemple de scénario

Cet exemple montre comment définir les trois propriétés de connexion réseau.

- Le groupe de répliques de conteneurs de notre flotte comprend 1 conteneur, qui exécute les processus du serveur de jeu. La confirmation d'exécution indique au conteneur d'exécuter 10 processus de serveur de jeu simultanés.

  Dans la définition du groupe de conteneurs de répliques, nous avons défini le `PortConfiguration` paramètre de ce conteneur comme suit :

  ```
  "PortConfiguration": {
    "ContainerPortRanges": [ { "FromPort": 10, "ToPort": 20, "Protocol": "TCP"} ]
    }
  ```

- Notre flotte possède également un groupe de conteneurs daemon avec 1 conteneur. Il a 1 processus qui nécessite un accès au réseau. Dans la définition du groupe de conteneurs daemon, nous avons défini le `PortConfiguration` paramètre de ce conteneur comme suit :

  ```
  "PortConfiguration": {
    "ContainerPortRanges": [ { "FromPort": 25, "ToPort": 25, "Protocol": "TCP"} ]  }
  ```

- Notre flotte est configurée avec 3 groupes de conteneurs répliques par instance de flotte. À partir de ces informations, nous pouvons utiliser la formule pour calculer le nombre de ports de connexion dont nous avons besoin :
  - Minimum : 31 ports [10 ports de conteneurs répliques* 3 groupes de conteneurs de répliques par instance + 1 port de conteneur de daemon]
  - Meilleure pratique : 62 ports [ports minimum* 2]

Lors de la création de la flotte de conteneurs, nous avons défini le `ConnectionPortRange` paramètre `ContainerGroupsConfiguration` comme suit :

```
"ConnectionPortRange": { "FromPort": 1010, "ToPort": 1071 }
```

- Nous voulons autoriser l'accès à tous les ports de connexion disponibles. Lors de la création de la flotte de conteneurs, nous avons défini le `EC2InboundPermissions` paramètre comme suit :

```
"EC2InboundPermissions": [
  {"FromPort": 1010, "ToPort": 1071, "IpRange": "10.24.34.0/23", "Protocol":
 "TCP"} ]
```

Configurez des contrôles de santé pour les conteneurs

Un conteneur redémarre automatiquement en cas de panne du terminal et cesse de fonctionner. Si le conteneur est essentiel, l'ensemble du groupe de conteneurs redémarre.

Vous pouvez définir des critères personnalisés supplémentaires pour mesurer l'état du conteneur et utiliser un bilan de santé pour tester ces critères. Pour configurer un contrôle de l'état d'un conteneur, vous pouvez le définir dans une image de conteneur docker ou dans la définition de votre conteneur. Si vous définissez un contrôle de santé dans la définition du conteneur, celui-ci remplace tous les paramètres de l'image du conteneur.

Définissez les contrôles de santé facultatifs en fonction du type de conteneur comme suit :

- Pour un conteneur de répliques essentiel, ne configurez pas les contrôles de santé. L' GameLiftagent Amazon gère automatiquement les rapports sur l'état de santé de ce conteneur.
- Pour les conteneurs de répliques non essentiels et les conteneurs de démons, vous pouvez éventuellement définir des paramètres de contrôle de santé.

Définissez les `ContainerDefinition` propriétés suivantes pour le contrôle de l'état d'un conteneur :

- `Command`— Fournissez une commande qui vérifie certains aspects de l'état du conteneur. C'est vous qui décidez des critères à utiliser pour mesurer l'état de santé. La commande doit générer une valeur de sortie de 1 (malsain) ou 0 (sain).

- `StartPeriod`— Spécifiez un délai initial avant que les échecs du bilan de santé ne commencent à être comptabilisés. Ce délai donne au conteneur le temps de démarrer ses processus.
- `Interval`— Décidez à quelle fréquence exécuter la commande de contrôle de santé. À quelle vitesse souhaitez-vous détecter et résoudre une panne de conteneur ?
- `Timeout`— Décidez du temps à attendre en cas de succès ou d'échec avant de réessayer la commande de contrôle de santé. Combien de temps la commande de contrôle de santé doit-elle prendre pour s'exécuter ?
- `Retries`— Combien de fois la commande de contrôle de santé doit-elle être réessayée avant d'enregistrer un échec ?

Définir les dépendances des conteneurs

Au sein de chaque groupe de conteneurs, vous pouvez définir des dépendances entre les conteneurs en fonction de l'état du conteneur. Une dépendance a un impact sur le moment où le conteneur dépendant peut démarrer ou s'arrêter en fonction de l'état d'un autre conteneur.

L'un des principaux cas d'utilisation des dépendances consiste à créer des séquences de démarrage et d'arrêt pour le groupe de conteneurs.

Par exemple, vous souhaiterez peut-être que le conteneur A démarre en premier et se termine correctement avant le démarrage des conteneurs B et C. Pour ce faire, créez d'abord une dépendance entre le conteneur B et le conteneur A, à condition que le conteneur A soit terminé avec succès. Créez ensuite une dépendance pour le conteneur C sur le conteneur A avec la même condition. Les séquences de démarrage se produisent dans l'ordre inverse de l'arrêt.

Configuration d'une flotte de conteneurs

Lorsque vous créez une flotte de conteneurs, prenez en compte les points de décision suivants. La plupart de ces points dépendent de l'architecture et de la configuration de votre conteneur.

Décidez où vous souhaitez déployer votre flotte

> En général, vous souhaitez déployer vos flottes géographiquement à proximité de vos joueurs afin de minimiser le temps de latence. Vous pouvez déployer votre flotte de conteneurs sur n'importe quel Each Région AWS pris GameLift en charge par Amazon. Si vous souhaitez déployer le même serveur de jeu dans d'autres zones géographiques, vous pouvez ajouter des sites distants à la flotte, notamment Régions AWS des Zones Locales. Dans le cas d'un parc multisite, vous pouvez ajuster la capacité indépendamment de chaque emplacement du parc.

Pour plus d'informations sur les emplacements de flotte pris en charge, consultezSites GameLift d'hébergement Amazon.

Choisissez un type et une taille d'instance pour votre flotte

Amazon GameLift prend en charge un large éventail de types d'instances Amazon EC2, qui sont tous disponibles pour une utilisation avec un parc de conteneurs. La disponibilité et le prix du type d'instance varient en fonction de l'emplacement. Vous pouvez consulter la liste des types d'instances pris en charge, filtrée par emplacement, dans la GameLift console Amazon (sous Ressources, Instance et quotas de service).

Lorsque vous choisissez un type d'instance, considérez d'abord la famille d'instances. Les familles d'instances offrent différentes combinaisons de capacités de processeur, de mémoire, de stockage et de mise en réseau. Obtenez plus d'informations sur les familles d'instances EC2. Au sein de chaque famille, vous avez le choix entre différentes tailles d'instance. Tenez compte des problèmes suivants lors de la sélection d'une taille d'instance :

- Quelle est la taille d'instance minimale capable de prendre en charge votre charge de travail ? Utilisez ces informations pour éliminer les types d'instances trop petits.
- Quelles tailles de type d'instance conviennent le mieux à votre architecture de conteneur ? Idéalement, vous devez choisir une taille qui puisse accueillir plusieurs copies de votre groupe de conteneurs de répliques avec un minimum d'espace perdu.
- Quelle granularité de mise à l'échelle convient le mieux à votre jeu ? L'augmentation de la capacité du parc implique l'ajout ou la suppression d'instances, et chaque instance représente la capacité d'héberger un nombre spécifique de sessions de jeu. Déterminez la capacité que vous souhaitez ajouter ou supprimer pour chaque instance. Si la demande des joueurs varie de plusieurs milliers d'une minute à l'autre, il peut être judicieux d'utiliser de très grandes instances pouvant héberger des centaines ou des milliers de sessions de jeu. En revanche, vous préférerez peut-être un contrôle de mise à l'échelle plus précis avec des types d'instances plus petits.
- Des économies sont-elles possibles en fonction de la taille ? Il se peut que le coût de certains types d'instances varie en fonction de la situation géographique en raison de la disponibilité.

Optimisez la configuration de votre environnement d'exécution

La configuration d'exécution d'une flotte est un ensemble d'instructions expliquant comment exécuter les processus du serveur pour l'hébergement de sessions de jeu. Ces instructions sont mises en œuvre par l' GameLift agent Amazon dans chaque groupe de répliques de conteneurs de la flotte.

La configuration d'exécution d'une flotte détermine le nombre de processus de serveur exécutés simultanément dans chaque groupe de conteneurs de répliques. Ce paramètre a un impact sur la façon dont vous calculez les limites de ressources de votre groupe de conteneurs et sur la manière dont vous choisissez un type d'instance pour votre flotte. Vous devez équilibrer ces trois éléments lors de la conception de votre flotte.

Pour plus d'informations sur l'utilisation des configurations d'exécution, consultezGérez la façon dont Amazon GameLift lance ses serveurs de jeux.

Définissez d'autres paramètres de flotte facultatifs

Vous pouvez utiliser les fonctionnalités optionnelles suivantes lors de la configuration d'une flotte de conteneurs :

- Configurez vos serveurs de jeu pour accéder à d'autres AWS ressources. veuillez consulter Communiquez avec les autres AWS ressources de vos flottes.
- Empêchez les sessions de jeu avec des joueurs actifs de se terminer prématurément lors d'un événement à réduction progressive.
- Limitez le nombre de sessions de jeu qu'une personne peut créer dans la flotte dans un laps de temps limité.

Création de définitions de groupes de conteneurs pour une flotte de GameLift conteneurs Amazon

Cette documentation concerne une fonctionnalité en version préliminaire publique. Elle est susceptible d’être modifiée.

Une définition de groupe de conteneurs décrit comment déployer vos applications de serveur de jeu conteneurisé sur une flotte de conteneurs. Il s'agit d'un plan qui identifie l'ensemble de conteneurs à utiliser dans la flotte et la manière de les gérer. Lorsque vous créez une flotte de conteneurs, vous spécifiez les définitions de groupes de conteneurs à déployer dans la flotte. Pour plus d'informations sur les groupes de conteneurs, consultezComposants du parc de conteneurs.

Avant de commencer

Réalisez les tâches suivantes :

- Concevez une architecture de conteneur pour héberger vos serveurs de jeux. veuillez consulter Concevez une flotte de GameLift conteneurs Amazon.

- Planifiez les définitions de conteneurs à inclure dans le groupe de conteneurs. Si vous utilisez la AWS CLI, créez votre définition de conteneur dans un fichier JSON.
- Transférez les images finales du conteneur vers un registre Amazon Elastic Container Registry (Amazon ECR) dans le Région AWS même registre que celui dans lequel vous prévoyez de créer le groupe de conteneurs. Amazon GameLift stocke un instantané de chaque image au moment où vous créez la définition du groupe de conteneurs, et utilise la copie lors du déploiement sur un parc de conteneurs. veuillez consulter Préparez une image de conteneur avec le logiciel de votre serveur de jeu.
- Vérifiez que votre AWS utilisateur dispose des autorisations IAM pour accéder au référentiel Amazon ECR. veuillez consulter Gérer les autorisations des utilisateurs pour Amazon GameLift. Au minimum, vous devez disposer d'autorisations pour effectuer les actions suivantes :
  - `ecr:DescribeImages`
  - `ecr:BatchGetImage`
  - `ecr:GetDownloadUrlForLayer`

Cloner une définition de groupe de conteneurs

Vous pouvez utiliser la GameLift console Amazon pour cloner une définition de groupe de conteneurs existante.

Pour cloner un groupe de conteneurs

1. Dans la GameLift console Amazon, accédez au volet de navigation de gauche et choisissez Container groups.
2. Sur la page de liste des groupes de conteneurs, sélectionnez le groupe de conteneurs existant que vous souhaitez cloner. Une fois que vous avez sélectionné un groupe de conteneurs, le bouton Cloner est actif.
3. Choisissez Clone (Cloner). Cette action ouvre l'assistant de création de groupes de conteneurs avec des paramètres préremplis.
4. Entrez un nouveau nom pour le groupe de conteneurs cloné. Les groupes de conteneurs d'une même région doivent avoir des noms uniques.
5. Parcourez les pages du groupe de conteneurs et de définition du conteneur, passez en revue et créez le nouveau groupe de conteneurs.

Création d'une définition de groupe de conteneurs de répliques

Un groupe de conteneurs répliques gère le logiciel de votre serveur de jeu. Un groupe de répliques de conteneurs possède au moins un conteneur qui exécute les processus Amazon GameLift Agent et de votre serveur de jeu. Le groupe peut disposer de conteneurs « annexes » supplémentaires pour exécuter les logiciels de support.

Cette rubrique explique comment créer une définition de groupe de conteneurs à l'aide de la GameLift console Amazon ou des outils AWS CLI. Pour des informations plus détaillées sur la définition des configurations de groupes de conteneurs, reportez-vous àConcevez une flotte de GameLift conteneurs Amazon.

Console

Dans la GameLift console Amazon, sélectionnez l' Région AWS endroit où vous souhaitez créer le groupe de conteneurs.

Ouvrez la barre de navigation gauche de la console et choisissez Groupes de conteneurs. Sur la page Groupes de conteneurs, choisissez Créer un groupe de conteneurs.

Étape 1 : Définissez les détails du groupe.

1. Entrez un nom de définition de groupe de conteneurs. Ce nom doit être propre à la région Compte AWS et. Dans la console, les définitions de groupes sont répertoriées par nom. Il peut donc être utile d'attribuer des libellés significatifs.
2. Sélectionnez la stratégie de planification des répliques.
3. Pour Limite de mémoire totale, entrez la mémoire maximale disponible pour le groupe de conteneurs. Pour obtenir de l'aide sur le calcul de cette valeur, consultezDéfinissez des limites de ressources.
4. Pour Limite totale du processeur, entrez la puissance de calcul maximale disponible pour le groupe de conteneurs. Pour obtenir de l'aide sur le calcul de cette valeur, consultezDéfinissez des limites de ressources.

Étape 2 : Ajoutez des définitions de conteneurs.

Définissez le conteneur avec votre application de serveur de jeu et l' GameLift agent Amazon. Il s'agit de votre réplique de conteneur essentielle.

1. Fournissez un nom de définition du conteneur. Chaque conteneur défini pour le groupe doit avoir une valeur de nom unique.
2. Identifiez l'URI de l'image Amazon ECR de l'image du conteneur. Entrez l'un des formats suivants :

   - URI de l'image uniquement : `[Compte AWS].dkr.ecr.[Région AWS].amazonaws.com/[repository ID]`
   - URI de l'image et résumé : `[Compte AWS].dkr.ecr.[Région AWS].amazonaws.com/[repository ID]@[digest]`
   - URI de l'image et balise : `[Compte AWS].dkr.ecr.[Région AWS].amazonaws.com/[repository ID]:[tag]`
3. Pour le conteneur essentiel, Oui est automatiquement sélectionné pour la première définition de conteneur. Si vous ajoutez une autre définition de conteneur, vous pouvez activer ou désactiver ce paramètre pour chaque définition. Pour en savoir plus, consultez Désignez les contenants essentiels.
4. Définissez une ou plusieurs plages de ports de conteneurs internes. Ce conteneur héberge vos serveurs de jeu. Définissez donc une plage avec suffisamment de ports pour que chaque processus serveur s'exécute dans le groupe de conteneurs. Pour en savoir plus, consultez Configuration des connexions réseau.
5. Les paramètres facultatifs Overrides et Environment variables vous permettent de spécifier les valeurs à transmettre au conteneur lors du lancement. Les valeurs que vous définissez ici remplacent les paramètres déjà présents dans l'image du conteneur.
6. Définissez des limites de conteneur facultatives pour gérer l'allocation des ressources pour ce conteneur. Pour en savoir plus, consultez Définissez des limites de ressources.
7. Définissez des contenants non essentiels supplémentaires selon les besoins :

   - Fournissez le nom de définition du conteneur et l'URI de l'image ECR. Les conteneurs non essentiels ne doivent pas exécuter l' GameLift agent Amazon.
   - Définissez une plage de ports de conteneurs internes uniquement si les conteneurs ont des processus nécessitant un accès au réseau.
   - Configurez éventuellement un bilan de santé pour le conteneur. Lorsqu'un conteneur non essentiel échoue à un contrôle de santé, seul le conteneur défaillant est redémarré.
   - Définissez éventuellement les remplacements, les variables d'environnement et les limites d'allocation de ressources selon les besoins.

Étape 3 : configurer les dépendances

Si vous avez plusieurs conteneurs dans votre définition de groupe de conteneurs, vous pouvez définir des dépendances entre eux. Utilisez les dépendances pour configurer les séquences de démarrage et d'arrêt en fonction de l'état du conteneur. Pour en savoir plus, consultez Définir les dépendances des conteneurs.

1. Identifiez le nom du conteneur pour lequel vous souhaitez ajouter une dépendance. Ce conteneur ne démarre pas tant que la condition de dépendance n'est pas satisfaite.
2. Identifiez le nom et la condition du conteneur de dépendance. Ce conteneur doit satisfaire à cette condition pour que le conteneur dépendant puisse démarrer.
3. Définissez des dépendances supplémentaires selon vos besoins. Vous pouvez créer plusieurs dépendances pour n'importe quel conteneur. Évitez de créer des dépendances circulaires.

Étape 4 : Révisez et créez.

1. Passez en revue tous les paramètres de définition de votre groupe de conteneurs. Vous ne pouvez pas modifier la configuration d'une définition de groupe de conteneurs une fois celle-ci créée. Utilisez Modifier pour apporter des modifications à n'importe quelle section, y compris à chacune de vos définitions de conteneur pour le groupe.
2. Lorsque vous avez terminé de réviser, choisissez Créer.

   Si votre demande aboutit, la console affiche la page détaillée de la nouvelle ressource de définition de groupe de conteneurs. Au départ`COPYING`, le statut est le suivant : Amazon GameLift commence à prendre des instantanés de toutes les images du conteneur pour le groupe. Lorsque cette phase est terminée, le statut de définition du groupe de conteneurs passe à`READY`. Une définition de groupe de conteneurs doit avoir le `READY` statut requis pour que vous puissiez créer une flotte de conteneurs à l'aide de cette définition.

AWS CLI

Lorsque vous utilisez la AWS CLI pour créer une définition de groupe de conteneurs, conservez vos configurations de définition de conteneur dans un `JSON` fichier séparé. Vous pouvez référencer le fichier dans votre commande CLI. Voir Création d'un `JSON` fichier de définition de conteneur pour des exemples de schéma.

Création d'une définition de groupe de conteneurs

Pour créer une nouvelle définition de groupe de conteneurs, utilisez la commande `create-container-group-definition` CLI. Pour plus d'informations sur cette commande, consultez le manuel create-container-group-definitionde référence des commandes de la AWS CLI.

Example : groupe de conteneurs de répliques

Cet exemple illustre une demande de définition de groupe de conteneurs de répliques. La structure de commande pour créer des définitions de répliques et de groupes de démons est essentiellement identique. Les détails spécifiques à chaque type de groupe sont décrits dans les définitions des conteneurs individuels.

Cet exemple suppose que vous avez créé un fichier JSON contenant les définitions de conteneur pour ce groupe.

```
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
```

Création d'un **JSON** fichier de définition de conteneur

Lorsque vous créez une définition de groupe de conteneurs, vous définissez également les conteneurs du groupe. Une définition de conteneur spécifie le référentiel Amazon ECR dans lequel l'image du conteneur est stockée, ainsi que les configurations facultatives pour les ports réseau, les limites d'utilisation du processeur et de la mémoire, ainsi que d'autres paramètres. Nous vous recommandons de créer un `JSON` fichier unique contenant les configurations de tous les conteneurs d'un groupe de conteneurs. La gestion d'un fichier est utile pour le stockage, le partage et le suivi des versions de ces configurations critiques. Si vous utilisez la AWS CLI pour créer vos définitions de groupes de conteneurs, vous pouvez référencer le fichier dans la commande.

Pour créer une définition de conteneur

1. Créez et ouvrez un nouveau `.JSON` fichier. Par exemple :

```
[~/work/glc]$ vim SimpleServer.json
```

2. Créez une définition de conteneur distincte pour chacun des conteneurs du groupe. Copiez l'exemple de contenu suivant et modifiez-le selon les besoins de vos conteneurs. Pour plus de détails sur la syntaxe d'une définition de conteneur, consultez ContainerDefinitionInputle Amazon GameLift API Reference.
3. Enregistrez le fichier localement afin de pouvoir y faire référence dans une commande AWS CLI.

Exemple : définition essentielle du conteneur de répliques

Example

Cet exemple décrit le conteneur essentiel pour votre groupe de répliques de conteneurs. Le conteneur de répliques essentiel inclut votre application de serveur de jeu, l'Amazon GameLift Agent, et peut inclure d'autres logiciels de support pour l'hébergement de votre jeu. La définition doit inclure un nom, une URI d'image et une configuration de port. Cet exemple définit également certaines limites de ressources spécifiques au conteneur.

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

Création d'une flotte de GameLift conteneurs Amazon

Cette documentation concerne une fonctionnalité en version préliminaire publique. Elle est susceptible d'être modifiée.

Lorsque vous avez créé vos définitions de groupes de conteneurs, utilisez la GameLift console Amazon ou le AWS Command Line Interface (AWS CLI) pour créer un parc de conteneurs.

Une fois que vous avez créé une nouvelle flotte, le statut de la flotte passe par plusieurs étapes au fur et à mesure qu'Amazon GameLift déploie vos groupes de conteneurs sur chaque instance de flotte et démarre les serveurs de jeu. Lorsque la flotte atteint son statut`ACTIVE`, elle est prête à accueillir des sessions de jeu. Pour obtenir de l'aide face à des problèmes de création de flotte, veuillez consulter Résoudre les problèmes liés à la GameLift flotte Amazon.

Console

Dans la GameLift console Amazon, sélectionnez l' Région AWS endroit où vous souhaitez créer la flotte. Les définitions des groupes de conteneurs doivent se trouver dans la même région que celle où vous souhaitez créer la flotte.

Ouvrez la barre de navigation gauche de la console et choisissez Fleets. Sur la page Flottes, choisissez Create fleet.

Étape 1 : Choisissez le type de calcul

- Choisissez le type de calcul Containers.

Étape 2 : définir les détails de la flotte

1. Dans la section Détails de la flotte, entrez le nom et la description de la flotte.
2. Dans la section Détails des groupes de conteneurs, identifiez les groupes de conteneurs à déployer dans la flotte. Vous devez ajouter un groupe de conteneurs de répliques. Vous pouvez éventuellement ajouter un groupe de conteneurs de démons. Chaque groupe doit avoir un statut`READY`.
3. Définissez la plage de ports de connexion pour le parc. Pour en savoir plus, consultez Configuration des connexions réseau.
4. Spécifiez éventuellement les répliques souhaitées par instance à déployer. Vous pouvez spécifier le nombre souhaité ou laisser Amazon GameLift calculer le nombre maximum possible. Si vous spécifiez un nombre souhaité supérieur au maximum calculé, la création de

flotte échouera. Vous ne pouvez pas modifier ce paramètre une fois la flotte créée. Pour plus de détails sur l'emballage groupé de répliques de conteneurs, consultezConcepts de base.

5. (Facultatif) Sous Informations supplémentaires :

   a. Pour Rôle d'instance, spécifiez un rôle IAM qui autorise les applications de votre version de jeu à accéder aux autres AWS ressources de votre compte. Pour plus d'informations, consultez Communiquez avec les autres AWS ressources de vos flottes. Pour créer une flotte avec un rôle d'instance, votre compte doit disposer de l'`PassRole`autorisation IAM. Pour plus d'informations, consultez Exemples d'autorisations IAM pour Amazon GameLift.

   b. Pour Groupe de mesures, entrez le nom d'un groupe de mesures de flotte nouveau ou existant. Vous pouvez agréger les mesures de plusieurs flottes en les ajoutant au même groupe de mesures.

Étape 3 : définir les détails de l'instance

1. Dans Déploiement d'instances, sélectionnez un ou plusieurs sites distants sur lesquels déployer les instances. La région d'origine est automatiquement sélectionnée (il s'agit de la région dans laquelle vous créez la flotte). Si vous sélectionnez des sites supplémentaires, des instances de flotte sont également déployées sur ces sites.

   > **Important**
   >
   > Pour utiliser des régions qui ne sont pas activées par défaut, activez-les dans votre Compte AWS.
   >
   > - Les flottes dont les régions ne sont pas activées et que vous avez créées avant le 28 février 2022 ne sont pas affectées.
   > - Pour créer de nouvelles flottes multi-sites ou pour mettre à jour les flottes multi-sites existantes, activez d'abord les régions que vous choisissez d'utiliser.
   >
   > Pour plus d'informations sur les régions qui ne sont pas activées par défaut et sur la manière de les activer, consultez la section Gestion Régions AWS dans le Références générales AWS.

2. Sélectionnez une configuration d'instance pour le parc. La console calcule automatiquement le minimum de vCPU et de mémoire requis (en fonction des limites totales que vous

définissez pour chaque groupe de conteneurs). Il filtre la liste complète des types d'instances disponibles en fonction des besoins en ressources et des emplacements que vous avez saisis. Vous pouvez ajouter des filtres supplémentaires selon vos besoins.

Pour plus d'informations sur le choix d'un type d'instance, consultezConfiguration d'une flotte de conteneurs. La taille du type d'instance que vous choisissez aura un impact sur la manière dont les groupes de conteneurs de répliques sont regroupés dans chaque instance de flotte. En fonction de votre choix, pensez à revoir vos paramètres pour les répliques souhaitées par instance.

Étape 4 : Configuration de l'environnement d'exécution

La configuration d'exécution détermine la manière dont les processus du serveur de jeu sont démarrés et exécutés. Ces instructions sont transmises à l' GameLift agent Amazon, qui les implémente de la même manière dans chaque groupe de conteneurs de répliques. Vous pouvez mettre à jour la configuration d'exécution d'une flotte en appelant UpdateRuntimeConfiguration.

1. Dans le champ Chemin de lancement, entrez le chemin d'accès à un exécutable de jeu.
2. (Facultatif) Dans Paramètres de lancement, entrez les informations à transmettre à l'exécutable de votre jeu sous forme d'ensemble de paramètres de ligne de commande.
3. Spécifiez le nombre de processus simultanés à maintenir en cours d'exécution dans chaque groupe de conteneurs de répliques. Consultez les GameLift quotas Amazon relatifs au nombre de processus serveur par instance. Les limites sur les processus serveur simultanés par instance s'appliquent au nombre total de processus simultanés pour toutes les configurations. Si vous configurez le parc de manière à dépasser la limite, le parc ne pourra pas être activé.
4. Définissez des limites facultatives pour les activations simultanées de sessions de jeu. Ces paramètres vous permettent de limiter la quantité de ressources consommées lors du démarrage d'une nouvelle session de jeu. Les activations de sessions de jeu peuvent avoir un impact sur les performances des sessions de jeu existantes.
5. Définissez les paramètres du port EC2 pour permettre au trafic externe d'accéder aux processus exécutés sur le parc. Spécifiez certains ou tous les numéros de ports de connexion définis pour le parc. Vous n'êtes pas obligé de définir ces ports lorsque vous créez la flotte, mais sans eux, aucun trafic ne peut se connecter à vos serveurs de jeu. Pour mettre à jour les paramètres de port d'une flotte ultérieurement, appelez UpdateFleetPortSettings

6. Sous Paramètres des ressources de session de jeu, configurez les fonctionnalités facultatives suivantes :

   a. Activez ou désactivez la politique de protection du dimensionnement du jeu. Lorsque la protection est activée, Amazon n' GameLift arrêtera pas les instances lors d'un événement de réduction si l'entreprise héberge une session de jeu active.

   b. Définissez une limite maximale de création de ressources pour limiter le nombre de sessions de jeu qu'un joueur peut créer pendant une période spécifiée.

Étape 5 : Configuration des balises

- (Facultatif) Ajoutez des balises au build en saisissant des paires clé et valeur. Choisissez Suivant pour passer à l'examen de la création de flotte.

Étape 6 : Révisez et créez.

- Passez en revue les paramètres de configuration de votre flotte.

  Vous pouvez mettre à jour les métadonnées et la configuration de la flotte à tout moment, quel que soit l'état de la flotte. Pour plus d'informations, consultez Gérez vos GameLift flottes Amazon. Vous pouvez mettre à jour la capacité de la flotte une fois que celle-ci a atteint le statut ACTIF. Pour plus d'informations, consultez Élargir la capacité GameLift d'hébergement d'Amazon. Vous pouvez également ajouter ou supprimer des sites distants.

  Lorsque vous avez terminé de réviser, choisissez Créer.

  Si votre demande est acceptée, la console affiche la page détaillée de la nouvelle ressource de flotte. Au départ`NEW`, le statut est le suivant : Amazon GameLift lance le processus de création de flotte. Vous pouvez suivre le statut de la nouvelle flotte sur la page Flottes. Une flotte est prête à accueillir des sessions de jeu lorsqu'elle atteint son statut`ACTIVE`.

AWS CLI

Pour créer une flotte de conteneurs avec le AWS CLI, ouvrez une fenêtre de ligne de commande et utilisez la `create-fleet` commande. Pour plus d'informations sur cette commande, consultez `create-fleet`la référence des AWS CLI commandes.

L'exemple de `create-fleet` demande ci-dessous crée une nouvelle flotte de conteneurs présentant les caractéristiques suivantes :

- ContainerGroupsConfiguration Spécifie une définition de groupe de conteneurs de répliques unique :`MegaFrogRaceServer.NA.v2`. Trois copies du groupe de répliques seront déployées sur chaque instance de flotte. Chaque instance dispose de 30 ports de connexion disponibles pour accéder aux processus de l'instance.
- La flotte utilise des instances à la demande c5.large.
- Il déploie des groupes de conteneurs aux emplacements suivants :
  - us-west-2 (région d'origine)
  - ca-central-1 (site distant)
- Chaque groupe de conteneurs de répliques d'une instance exécutera 5 processus de serveur de jeu simultanément, ce qui permettra à chaque instance d'héberger jusqu'à 15 sessions de jeu à la fois.
- Dans chaque groupe de répliques de conteneurs, Amazon GameLift autorise l'activation simultanée de deux nouvelles sessions de jeu. Cela met également fin à toute session de jeu en cours d'activation s'ils ne sont pas prêts à héberger des joueurs dans les 300 secondes.
- La protection de session de jeu sera activée pour toutes les sessions de jeu hébergées sur les instances de cette flotte.
- les joueurs individuels peuvent créer trois nouvelles sessions de jeu au cours d'une période de 15 minutes.

```
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" \
```

Si la demande de création de flotte aboutit, Amazon GameLift renvoie un ensemble d'attributs de flotte comprenant les paramètres de configuration que vous avez demandés et un nouvel identifiant de flotte. Amazon définit GameLift ensuite le statut de la flotte et les statuts de localisation sur Nouveau et lance le processus d'activation de la flotte. Vous pouvez suivre le statut de la flotte et consulter d'autres d'informations de flotte à l'aide de ces commandes d'interface de ligne de commande :

- 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

Vous pouvez changer la capacité de la flotte et d'autres paramètres de configuration, si nécessaire, à l'aide des commandes suivantes :

- update-fleet-attributes
- update-fleet-capacity
- update-fleet-port-settings
- update-runtime-configuration
- create-fleet-locations
- delete-fleet-locations

Gérez vos flottes de GameLift conteneurs Amazon

Cette documentation concerne une fonctionnalité en version préliminaire publique. Elle est susceptible d'être modifiée.

Lorsque vous souhaitez obtenir des informations sur votre flotte de conteneurs ou apporter des modifications, vous pouvez utiliser les actions suivantes pour gérer votre flotte de conteneurs.

Afficher les ressources

Voici quelques moyens d'obtenir des informations sur les ressources de votre flotte de conteneurs.

- DescribeCompute- Renvoie les détails d'un conteneur enregistré en tant que calcul.
- DescribeContainerGroupDefinition- Renvoie des informations sur la définition d'un groupe de conteneurs. Cette ressource décrit comment le groupe et ses conteneurs sont configurés.
- DescribeFleetAttributes- Obtient les attributs de la flotte, notamment la plage de ports de connexion, et d'autres attributs.
- DescribeFleetCapacity- Décompte les groupes de répliques de conteneurs de la flotte et leur statut.
- DescribeRuntimeConfiguration- Décrit les processus du serveur qui s'exécutent dans chaque groupe de conteneurs de répliques.
- GetComputeAccess- Fournit un accès à distance à une instance hébergeant le groupe de conteneurs.
- GetComputeAuthToken- Demande un jeton d'authentification à Amazon GameLift pour une ressource de calcul dans un parc de conteneurs.
- ListCompute- Répertorie les groupes de conteneurs enregistrés sous forme de calculs.
- ListContainerGroupDefinitions- Répertorie les définitions des groupes de conteneurs.
- ListFleets- Répertorie les flottes utilisant un groupe de conteneurs spécifique.

Mettre à jour les ressources

Voici quelques moyens de créer et de modifier des ressources de flotte de conteneurs.

- CreateContainerGroupDefinition- Crée une définition de groupe de conteneurs.
- CreateFleet- Crée une flotte de conteneurs lorsque `ComputeType` ce paramètre est défini sur`CONTAINER`.
- RegisterCompute- Enregistre les ordinateurs d'une flotte de conteneurs.

- UpdateFleetAttributes- Met à jour les attributs modifiables d'une flotte, tels que les options de configuration de flotte Anywhere.
- UpdateFleetCapacity- Met à jour les paramètres de capacité d'un parc EC2 géré ou d'un parc de conteneurs.
- UpdateRuntimeConfiguration- Met à jour la configuration d'exécution, qui décrit les processus du serveur à exécuter dans chaque groupe de conteneurs de répliques enregistré en tant que calcul.

Supprimer des ressources

Voici quelques moyens de supprimer les ressources du parc de conteneurs.

- DeleteContainerGroupDefinition- Supprime une définition de groupe de conteneurs.
- DeleteFleet- Supprime une flotte.
- DeregisterCompute- Supprime une ressource informatique d'un parc de conteneurs.

Dimensionnement des flottes de GameLift conteneurs Amazon

Cette documentation concerne une fonctionnalité en version préliminaire publique. Elle est susceptible d'être modifiée.

L'une des tâches les plus difficiles de l'hébergement de jeux consiste à augmenter la capacité pour répondre à la demande des joueurs sans gaspiller de l'argent en ressources dont vous n'avez pas besoin. Dans une flotte de conteneurs, vous augmentez la capacité de votre flotte en ajoutant ou en supprimant des instances de flotte.

Lorsque vous créez une nouvelle flotte, Amazon GameLift définit la capacité souhaitée de la flotte à une instance et déploie une instance dans la région d'origine de la flotte. Pour un parc multi-sites, Amazon GameLift déploie une instance dans la région d'origine et sur chaque site distant. Une fois que l'état de la flotte est atteint`ACTIVE`, vous pouvez augmenter la capacité souhaitée pour augmenter ou réduire la capacité souhaitée pour la réduire.

Vous pouvez utiliser les fonctionnalités de GameLift dimensionnement d'Amazon pour modifier la capacité manuellement ou configurer le dimensionnement automatique en fonction de la demande des joueurs :

- Configurez le dimensionnement automatique avec le suivi des cibles. veuillez consulter Mise à l'échelle automatique basée sur les cibles.

- Modifiez manuellement la capacité de votre flotte. veuillez consulter Configuration manuelle de la capacité d'une GameLift flotte Amazon.

Lorsque vous agrandissez une flotte de conteneurs, réfléchissez à l'impact de l'ajout ou de la suppression d'instances sur la capacité de la flotte à accueillir des sessions de jeu et des joueurs.

- Sessions de jeu par instance
  - Chaque processus de serveur de jeu exécuté sur une instance représente la capacité d'héberger une session de jeu.
  - Utilisez cette formule pour calculer le nombre de sessions de jeu exécutées simultanément sur une instance de flotte de conteneurs :

```
[Game sessions per instance] = [# of processes per replica container group] * [# of
 replica container groups per instance]
```

    - Pour les processus par groupe de conteneurs de répliques, appelez DescribeRuntimeConfigurationet comptez le nombre d'exécutions simultanées pour les processus du serveur de jeu.
    - Pour les groupes de conteneurs de répliques par instance, appelez DescribeFleetAttributespour obtenir la `DesiredReplicaContainerGroupPerInstance` valeur. Si cette valeur n'est pas définie, `MaxReplicaContainerGroupsPerInstance` utilisez-la.
- Joueurs par instance
  - Vous décidez du nombre de machines à sous à autoriser lors de chaque session de jeu. Selon la façon dont votre solution d'hébergement gère le placement des sessions de jeu, vous pouvez définir les joueurs par session de jeu dans votre configuration de matchmaking ou dans vos appels pour démarrer un placement de session de jeu.
  - Utilisez cette formule pour calculer le nombre de joueurs qui peuvent jouer à votre jeu simultanément sur une instance de flotte de conteneurs :

```
[Players per instance] = [# of game sessions per instance] * [# of player slots per
 game session]
```

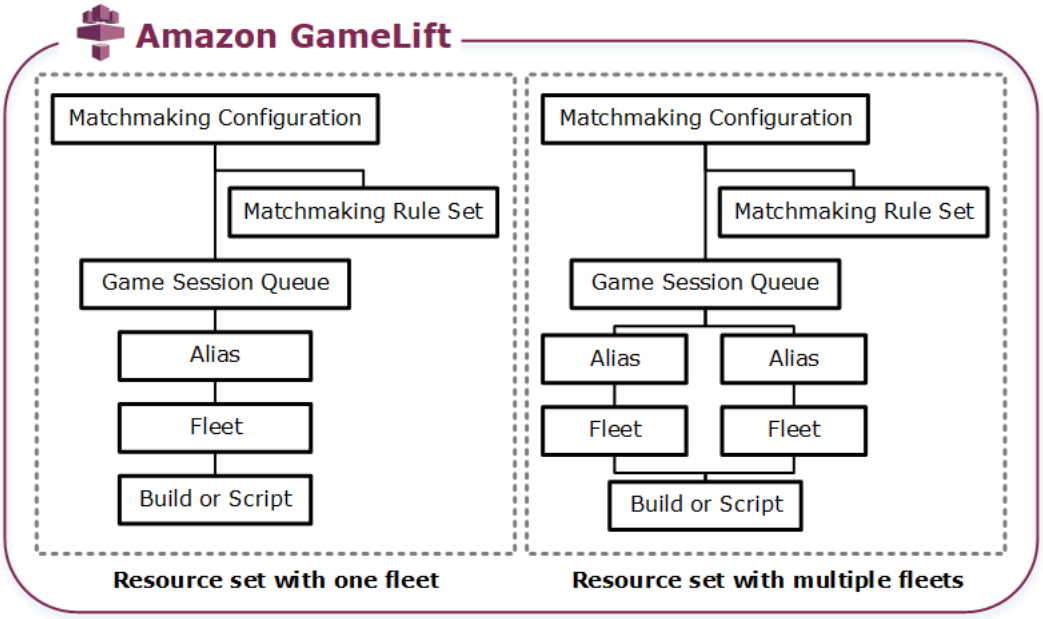
Pour connaître la capacité totale actuelle d'une flotte de conteneurs, appelez DescribeFleetCapacityou DescribeFleetLocation Capacity pour obtenir le nombre de répliques de

groupes de conteneurs de la flotte. Les groupes actifs sont ceux qui hébergent actuellement des sessions de jeu. Les groupes inactifs sont prêts à héberger une nouvelle session de jeu. Multipliez ces valeurs par le nombre de processus serveur par groupe de conteneurs de répliques.

Gestion des ressources GameLift d'hébergement Amazon

Cette section fournit des informations détaillées sur la configuration des ressources GameLift gérées par Amazon pour faire fonctionner vos serveurs de jeu et héberger des sessions de jeu pour les joueurs. Vous devez configurer et déployer des ressources, adapter la capacité à la demande des joueurs et localiser les ressources disponibles pour héberger des sessions de jeu.

Le schéma suivant montre comment les objets de GameLift ressources Amazon sont liés les uns aux autres. Utilisez un build ou un script pour créer une flotte, lui attribuer un alias et ajouter des flottes à une file d'attente de session de jeu en utilisant leur alias. Pour les jeux qui utilisent le FlexMatch matchmaking, utilisez la file d'attente des sessions de jeu et un ensemble de règles de matchmaking pour créer une configuration de matchmaking.



Code du serveur de jeu

- Build : votre logiciel de serveur de jeu personnalisé qui s'exécute sur Amazon GameLift et héberge des sessions de jeu pour vos joueurs. Une version de jeu représente l'ensemble de fichiers qui exécutent votre serveur de jeu sur un système d'exploitation particulier et que vous devez intégrer à AmazonGameLift. Téléchargez les fichiers de création de jeux sur Amazon GameLift à l'Régions AWSendroit où vous prévoyez de créer des flottes. Pour plus d'informations, veuillez consulter Téléchargez une version de serveur personnalisée sur Amazon GameLift.
- Script : votre configuration et votre logique de jeu personnalisées à utiliser avec les serveurs en temps réel. Configurez des serveurs en temps réel pour vos clients de jeu en créant un

script à l'aide JavaScript d'une logique de jeu personnalisée et ajoutez-y une logique de jeu personnalisée pour héberger des sessions de jeu pour vos joueurs. Pour plus d'informations, veuillez consulter Importer un script de serveurs en temps réel sur Amazon GameLift.

Parc

Ensemble de ressources informatiques qui font fonctionner vos serveurs de jeu et hébergent des sessions de jeu pour vos joueurs. Pour plus d'informations sur les endroits où vous pouvez déployer des flottes, consultezSites GameLift d'hébergement Amazon. Pour plus d'informations sur la création de flottes, consultezConfiguration des GameLift flottes Amazon.

Alias

Un identifiant abstrait pour une flotte que vous pouvez utiliser pour modifier à tout moment la flotte à laquelle vos joueurs sont connectés. Pour plus d'informations, veuillez consulter Ajouter un alias à une GameLift flotte Amazon.

File d'attente des sessions de jeu

Mécanisme de placement de sessions de jeu qui reçoit les demandes de nouvelles sessions de jeu et recherche les serveurs de jeu disponibles pour héberger les nouvelles sessions. Pour plus d'informations sur les files d'attente des sessions de jeu, consultezConfiguration des GameLift files d'attente Amazon pour le placement des sessions de jeu.

Téléchargement de versions et de scripts sur Amazon GameLift

Avant de déployer vos serveurs de jeu multijoueurs pour les héberger sur AmazonGameLift, vous devez charger les fichiers de vos serveurs de jeu. Les rubriques de cette section fournissent des instructions sur la préparation et le téléchargement de fichiers de construction de serveurs de jeu personnalisés ou de fichiers de script de serveur Realtime Servers.

Rubriques

- Téléchargez une version de serveur personnalisée sur Amazon GameLift
- Importer un script de serveurs en temps réel sur Amazon GameLift

Téléchargez une version de serveur personnalisée sur Amazon GameLift

Après avoir intégré votre serveur de jeu à Amazon GameLift, téléchargez les fichiers de compilation sur Amazon GameLift. Cette rubrique explique comment empaqueter les fichiers de build de votre

jeu, créer un script d'installation de build facultatif, puis télécharger les fichiers à l'aide du AWS Command Line Interface(AWS CLI) ou d'un AWS SDK.

Rubriques

Conditionner les fichiers de version de jeu

Avant de télécharger votre serveur de jeu configuré sur Amazon GameLift, empaquetez les fichiers de compilation du jeu dans un répertoire de compilation. Ce répertoire doit inclure tous les composants nécessaires pour l'exécution de vos serveurs de jeux et l'hébergement de sessions, y compris les éléments suivants :

- Fichiers binaires du serveur de jeu : fichiers binaires nécessaires au fonctionnement du serveur de jeu. Une version peut inclure des fichiers binaires pour plusieurs serveurs de jeu conçus pour fonctionner sur la même plateforme. Pour obtenir la liste des plateformes prises en charge, consultezAssistance au développement avec Amazon GameLift.
- Dépendances : tous les fichiers dépendants dont les exécutables de votre serveur de jeu ont besoin pour fonctionner. Exemples : actifs, fichiers de configuration et bibliothèques dépendantes.

  > **Note**
  >
  > Pour les versions de jeu créées avec le SDK Amazon GameLift Server pour C++ (y compris celles créées avec le plugin Unreal), incluez la DLL OpenSSL pour la même version d'OpenSSL que celle avec laquelle vous avez créé le SDK du serveur. Consultez le fichier README du SDK du serveur pour plus de détails.

- Script d'installation (facultatif) : fichier de script permettant de gérer les tâches d'installation de votre version de jeu sur les serveurs GameLift d'hébergement Amazon. Placez ce fichier à la racine du répertoire de construction. Amazon GameLift exécute le script d'installation dans le cadre de la création de la flotte.

Vous pouvez configurer n'importe quelle application de votre build, y compris votre script d'installation, pour accéder à vos ressources en toute sécurité sur d'autres AWS services. Pour plus d'informations sur les méthodes à suivre, consultezCommuniquez avec les autres AWS ressources de vos flottes.

Après avoir empaqueté vos fichiers de compilation, assurez-vous que votre serveur de jeu peut fonctionner sur une nouvelle installation de votre système d'exploitation cible. Cela permet de vérifier que vous incluez toutes les dépendances requises dans votre package et que votre script d'installation est correct.

Créez un GameLift build Amazon

Lorsque vous créez une build et téléchargez vos fichiers, vous disposez de quelques options :

- Création d'un build à partir d'un répertoire de fichiers. Il s'agit de l'option la plus simple et la plus couramment utilisée.
- Créez un build avec des fichiers dans Amazon Simple Storage Service (Amazon S3). Avec cette option, vous pouvez gérer vos versions de build dans Amazon S3.

Avec les deux méthodes, Amazon GameLift crée une nouvelle ressource de build avec un identifiant de build unique et d'autres métadonnées. La construction démarre avec le statut Initialisé. Une fois qu'Amazon a GameLift acquis les fichiers du serveur de jeu, le build passe à l'état Prêt.

Lorsque le build est prêt, vous pouvez le déployer sur une nouvelle GameLift flotte Amazon. Pour plus d'informations, consultez Créez une flotte GameLift gérée par Amazon .Quand Amazon GameLift configure le nouveau parc, il télécharge les fichiers de construction sur chaque instance de flotte et installe les fichiers de génération.

Création d'un build à partir d'un répertoire de fichiers

Pour créer une version de jeu stockée dans n'importe quel emplacement, y compris dans un répertoire local, utilisez la `upload-build`AWS CLIcommande. Cette commande crée un nouvel enregistrement de build dans Amazon GameLift et télécharge les fichiers à partir d'un emplacement que vous spécifiez.

Envoyez une demande de chargement. Dans une fenêtre de ligne de commande, entrez la upload-build commande et les paramètres suivants.

```
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— L'environnement d'exécution du build du serveur de jeu. Vous devez spécifier une valeur de système d'exploitation. Vous ne pourrez pas le mettre à jour ultérieurement.
- server-sdk-version— La version du SDK GameLift du serveur Amazon à laquelle votre serveur de jeu est intégré. Si vous ne fournissez aucune valeur, Amazon GameLift utilise la valeur par défaut`4.0.2`. Si vous spécifiez une version du SDK de serveur incorrecte, la compilation du serveur de jeu risque d'échouer lors de `InitSdk` l'appel pour établir une connexion au GameLift service Amazon.
- build-root— Le chemin du répertoire de vos fichiers de compilation.
- name— Nom descriptif de la nouvelle version.
- build-version— Les détails de version des fichiers de compilation.
- region— La AWS région dans laquelle vous souhaitez créer votre build. Créez le build dans la région où vous prévoyez de déployer des flottes. Si vous déployez votre jeu dans plusieurs régions, créez un build dans chaque région.

> **Note**
>
> Consultez votre région par défaut actuelle à l'aide du aws configure get region.
> Pour modifier votre région par défaut, utilisez la aws configure set region ***region name***commande.

Exemples

```
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
```

En réponse à votre demande de téléchargement, Amazon GameLift indique l'état d'avancement du téléchargement. En cas de téléchargement réussi, Amazon GameLift renvoie le nouveau numéro d'enregistrement de build. Le temps de chargement dépend de la taille de vos fichiers de jeu et de la vitesse de connexion.

Création d'un build avec des fichiers dans Amazon S3

Vous pouvez stocker vos fichiers de compilation dans Amazon S3 et les télécharger sur Amazon à GameLift partir de là. Lorsque vous créez votre build, vous spécifiez l'emplacement du compartiment S3, et Amazon GameLift récupère les fichiers de build directement depuis Amazon S3.

Pour créer une ressource de construction

1. Stockez vos fichiers de compilation dans Amazon S3. Créez un fichier .zip contenant les fichiers de construction empaquetés et téléchargez-le dans un compartiment S3 de votreCompte AWS. Prenez note de l'étiquette du compartiment et du nom du fichier, vous en aurez besoin lors de la création d'une GameLift version Amazon.
2. Donnez à Amazon GameLift l'accès à vos fichiers de compilation. Créez un rôle IAM en suivant les instructions deAccédez à un fichier de compilation de jeu dans Amazon S3. Après avoir créé le rôle, prenez note du nom de ressource Amazon (ARN) du nouveau rôle, dont vous aurez besoin lors de la création d'un build.
3. Créez un build. Utilisez la GameLift console Amazon ou le AWS CLI pour créer un nouvel enregistrement de build. Vous devez avoir l'`PassRole`autorisation, comme décrit dansExemples d'autorisations IAM pour Amazon GameLift.

Console

1. Dans la GameLiftconsole Amazon, dans le volet de navigation, choisissez Hosting, Builds.
2. Sur la page Builds, choisissez Create build.
3. Sur la page Créer une version, sous Paramètres de génération, procédez comme suit :

a. Dans Nom, entrez le nom du script.

b. Pour Version, entrez une version. Comme vous pouvez mettre à jour le contenu d'une version, les données de version peuvent vous aider à suivre les mises à jour.

c. Dans Système d'exploitation (OS), choisissez le système d'exploitation de la version de votre serveur de jeu. Vous ne pourrez pas mettre à jour cette valeur ultérieurement.

d. Pour le build du serveur de jeu, entrez l'URI S3 de l'objet de build que vous avez chargé sur Amazon S3, puis choisissez la version de l'objet. Si vous ne vous souvenez pas de l'URI et de la version de l'objet Amazon S3, choisissez Browse S3 et recherchez l'objet de construction.

e. Pour le rôle IAM, choisissez le rôle que vous avez créé qui permet à Amazon GameLift d'accéder à votre compartiment S3 et à votre objet de construction.

4. (Facultatif) Sous Balises, ajoutez des balises au build en saisissant des paires clé et valeur.

5. Sélectionnez Create (Créer).

Amazon GameLift attribue un identifiant à la nouvelle version et télécharge le fichier .zip désigné. Vous pouvez consulter le nouveau build, y compris son statut, sur la page Builds.

AWS CLI

Pour définir le nouveau build et télécharger les fichiers de build de votre serveur, utilisez la `create-build`commande.

1. Ouvrez une fenêtre de ligne de commande et passez à un répertoire dans lequel vous pouvez utiliser leAWS CLI.

2. Entrez la create-build commande suivante :

```
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— Nom descriptif de la nouvelle version.

- server-sdk-version— La version du SDK du GameLift serveur Amazon que vous avez utilisée pour intégrer votre serveur de jeu à Amazon GameLift. Si vous ne fournissez aucune valeur, Amazon GameLift utilise la valeur par défaut`4.0.2`.
- operating-system— L'environnement d'exécution du build du serveur de jeu. Vous devez spécifier une valeur de système d'exploitation. Vous ne pourrez pas le mettre à jour ultérieurement.
- build-version— Les détails de version des fichiers de compilation. Ces informations peuvent être utiles car chaque nouvelle version de votre serveur de jeu nécessite une nouvelle ressource de compilation.
- storage-location
  - Bucket— Le nom du compartiment S3 qui contient votre build. Par exemple, « my_build_files ».
  - Key— Le nom du fichier .zip qui contient vos fichiers de compilation. Par exemple, « my_game_build_7.0.1, 7.0.2 ».
  - RoleARN— L'ARN attribué au rôle IAM que vous avez créé. Par exemple, « arn:aws:iam : :111122223333:role/ ». GameLiftAccess Pour un exemple de politique, consultez Accédez à un fichier de compilation de jeu dans Amazon S3.
- region— Créez le build dans la AWS région où vous prévoyez de déployer des flottes. Si vous déployez votre jeu dans plusieurs régions, créez un build dans chaque région.

> **Note**
>
> Nous vous recommandons de vérifier votre région par défaut actuelle à l'aide de la `configure get`commande . Pour modifier votre région par défaut, utilisez la `configure set`commande.

Example (Exemple)

```
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. Pour afficher la nouvelle version, utilisez la `describe-build`commande.

Mettez à jour vos fichiers de compilation

Vous pouvez mettre à jour les métadonnées d'une ressource de build à l'aide de la GameLift console Amazon ou de la `update-build`AWS CLIcommande.

Une fois que vous avez créé un GameLift build Amazon, vous ne pouvez pas mettre à jour les fichiers de build qui y sont associés. Pour chaque nouvel ensemble de fichiers, créez un nouveau GameLift build Amazon. À l'aide de cette `upload-build`commande, Amazon crée GameLift automatiquement un nouvel enregistrement de build pour chaque demande. Si vous fournissez des fichiers de compilation à l'aide de la `create-build`commande, chargez un nouveau fichier .zip portant un nom différent sur Amazon S3 et créez une version en faisant référence au nouveau nom de fichier.

Pour déployer des versions de génération mises à jour, voici quelques astuces :

- Utilisez les files d'attente et remplacez des flottes selon vos besoins. Lorsque vous configurez votre client de jeu avec Amazon GameLift, spécifiez une file d'attente plutôt qu'une flotte. Avec les files d'attente, vous pouvez ajouter les nouvelles flottes avec la nouvelle version à votre file d'attente et supprimer les anciennes flottes. Pour plus d'informations, veuillez consulter Configuration des GameLift files d'attente Amazon pour le placement des sessions de jeu.
- Utilisez des alias pour transférer les joueurs vers une nouvelle version du jeu. Lorsque vous intégrez votre client de jeu à Amazon GameLift, spécifiez un alias de flotte au lieu d'un identifiant de flotte. Pour plus d'informations, veuillez consulter Ajouter un alias à une GameLift flotte Amazon.
- Configurez des mises à jour de build automatisées. Pour obtenir des exemples de scripts et des informations sur l'intégration GameLift des déploiements Amazon dans votre système de compilation, consultez Automatiser les déploiements GameLift sur Amazon sur le blog AWS Game Tech.

Ajouter un script d'installation de build

Créez un script d'installation pour le système d'exploitation (OS) de votre build de jeu :

- Windows : créez un fichier batch nomméinstall.bat.
- Linux : créez un fichier de script shell nomméinstall.sh.

Lors de la création d'un script d'installation, gardez à l'esprit les informations suivantes :

- Le script ne peut accepter aucune entrée utilisateur.
- Amazon GameLift installe le build et recrée les répertoires de fichiers de votre package de build sur un serveur d'hébergement aux emplacements suivants :
  - Flottes Windows : `C:\game`
  - Flottes Linux : `/local/game`
- Pendant le processus d'installation des flottes Linux, l'utilisateur run-as dispose d'un accès limité à la structure des fichiers d'instance. Cet utilisateur dispose de tous les droits sur le répertoire dans lequel vos fichiers de compilation sont installés. Si votre script d'installation exécute des actions qui nécessitent des autorisations d'administrateur, spécifiez l'accès administrateur à l'aide desudo. L'utilisateur run-as pour les flottes Windows dispose d'autorisations d'administrateur par défaut. Les échecs d'autorisation liés au script d'installation génèrent un message d'événement indiquant un problème avec le script.
- Sur Linux, Amazon GameLift prend en charge les langages d'interprétation shell courants tels que bash. Ajoutez un package (par exemple, `#!/bin/bash`) en haut de votre script d'installation. Pour vérifier la prise en charge de vos commandes shell préférées, accédez à distance à une instance Linux active et ouvrez une invite du shell. Pour plus d'informations, veuillez consulter Connectez-vous à distance aux instances GameLift de flotte Amazon.
- Le script d'installation ne peut pas s'appuyer sur une connexion d'appairage VPC. Une connexion d'appairage VPC n'est pas disponible tant qu'Amazon n'a pas GameLift installé le build on fleet instances.

Example Fichier bash d'installation de Windows

Ce `install.bat` fichier d'exemple installe les composants d'exécution Visual C++ requis pour le serveur de jeu et écrit les résultats dans un fichier journal. Le script inclut le fichier de composant dans le package de construction à la racine.

```
vcredist_x64.exe /install /quiet /norestart /log c:\game\vcredist_2013_x64.log
```

Example Script shell d'installation de Linux

Ce `install.sh` fichier d'exemple utilise bash dans le script d'installation et écrit les résultats dans un fichier journal.

```
#!/bin/bash
```

```
echo 'Hello World' > install.log
```

Ce `install.sh` fichier d'exemple montre comment vous pouvez utiliser l' CloudWatch agent Amazon pour collecter des métriques personnalisées et au niveau du système, et gérer la rotation des journaux. Amazon étant GameLift exécuté dans un VPC de service, vous devez accorder à Amazon l' GameLift autorisation d'assumer un rôle AWS Identity and Access Management (IAM) en votre nom. Pour permettre GameLift à Amazon d'assumer un rôle, créez un rôle qui inclut la politique AWS `CloudWatchAgentAdminPolicy` gérée et utilisez ce rôle lorsque vous créez une flotte.

```
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
```

Importer un script de serveurs en temps réel sur Amazon GameLift

Lorsque vous êtes prêt à déployer des serveurs en temps réel pour votre jeu, chargez les fichiers de script de serveur en temps réel complets sur Amazon. GameLift Pour ce faire, créez une ressource de GameLift script Amazon et spécifiez l'emplacement de vos fichiers de script. Vous pouvez également mettre à jour les fichiers de script du serveur qui sont déjà déployés en chargeant de nouveaux fichiers pour une ressource de script existante.

Lorsque vous créez une nouvelle ressource de script, Amazon lui GameLift attribue un identifiant de script unique (par exemple,`script-1111aaaa-22bb-33cc-44dd-5555eeee66ff`) et télécharge une copie des fichiers de script. La durée du téléchargement dépend de la taille de vos fichiers de script et de la vitesse de votre connexion.

Une fois que vous avez créé la ressource de script, Amazon GameLift déploie le script avec une nouvelle flotte de serveurs en temps réel. Amazon GameLift installe le script de votre serveur sur chaque instance du parc et y place les fichiers de script. `/local/game`

Pour résoudre les problèmes d'activation du parc liés au script du serveur, reportez-vous Résoudre les problèmes liés à la GameLift flotte Amazon à la section.

Fichiers de script du package

Le script de votre serveur peut inclure un ou plusieurs fichiers combinés dans un seul fichier .zip pour le téléchargement. Le fichier .zip doit contenir tous les fichiers dont votre script a besoin pour s'exécuter.

Vous pouvez stocker vos fichiers de script compressés dans un répertoire de fichiers local ou dans un compartiment Amazon Simple Storage Service (Amazon S3).

Charger des fichiers de script depuis un répertoire local

Si vos fichiers de script sont stockés localement, vous pouvez les charger sur Amazon à GameLift partir de là. Pour créer la ressource de script, utilisez la GameLift console Amazon ou le AWS Command Line Interface(AWS CLI).

Amazon GameLift console

Pour créer une ressource de script

1. Ouvrez la GameLiftconsole Amazon.
2. Dans le volet de navigation, choisissez Hosting, Scripts.
3. Sur la page Scripts, choisissez Créer un script.
4. Sur la page Créer un script, sous Paramètres du script, procédez comme suit :
   a. Dans Nom, entrez un nom de script.
   b. (Facultatif) Pour Version, entrez les informations de version. Comme vous pouvez mettre à jour le contenu d'un script, les données de version peuvent être utiles pour le suivi des mises à jour.
   c. Pour Source du script, choisissez Charger un fichier .zip.
   d. Pour les fichiers de script, choisissez Choisir un fichier, recherchez le fichier .zip qui contient votre script, puis sélectionnez-le.
5. (Facultatif) Sous Balises, ajoutez des balises au script en saisissant des paires clé/valeur.
6. Sélectionnez Create (Créer).

   Amazon GameLift attribue un identifiant au nouveau script et télécharge le fichier .zip désigné. Vous pouvez consulter le nouveau script, y compris son statut, sur la page Scripts.

AWS CLI

Utilisez la `create-script`AWS CLIcommande pour définir le nouveau script et charger les fichiers de script de votre serveur.

Pour créer une ressource de script

1. Placez le fichier .zip dans un répertoire dans lequel vous pouvez utiliser leAWS CLI.

2. Ouvrez une fenêtre de ligne de commande et accédez au répertoire dans lequel vous avez placé le fichier .zip.
3. Entrez la create-script commande et les paramètres suivants. Pour le --zip-file paramètre, veillez à ajouter la chaîne `fileb://` au nom du fichier .zip. Il identifie le fichier comme étant binaire afin qu'Amazon GameLift traite le contenu compressé.

```
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
```

Example (Exemple)

```
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
```

En réponse à votre demande, Amazon GameLift renvoie le nouvel objet de script.

4. Pour voir le nouveau script, appelez`describe-script`.

Importer des fichiers de script depuis Amazon S3

Vous pouvez stocker vos fichiers de script dans un compartiment Amazon S3 et les charger sur Amazon à GameLift partir de ce compartiment. Lorsque vous créez votre script, vous spécifiez l'emplacement du compartiment S3 et Amazon GameLift récupère vos fichiers de script depuis Amazon S3.

Pour créer une ressource de script

1. Stockez vos fichiers de script dans un compartiment S3. Créez un fichier .zip contenant les fichiers de script de votre serveur et chargez-le dans un compartiment S3 dans un environnement Compte AWS que vous contrôlez. Prenez note de l'URI de l'objet : vous en avez besoin lors de la création d'un script AmazonGameLift.

> **Note**
>
> Amazon GameLift ne prend pas en charge le chargement à partir de compartiments S3 dont le nom contient un point (.).

2. Donnez à Amazon GameLift l'accès à vos fichiers de script. Pour créer un rôle AWS Identity and Access Management (IAM) permettant GameLift à Amazon d'accéder au compartiment S3 contenant votre script de serveur, suivez les instructions figurant dansConfigurer un rôle de service IAM pour Amazon GameLift. Après avoir créé le nouveau rôle, prenez note de son nom, dont vous aurez besoin lors de la création d'un script.
3. Créez un script. Utilisez la GameLift console Amazon ou le AWS CLI pour créer un nouvel enregistrement de script. Pour effectuer cette demande, vous devez disposer de l'`PassRole`autorisation IAM, comme décrit dansExemples d'autorisations IAM pour Amazon GameLift.

Amazon GameLift console

1. Dans la GameLiftconsole Amazon, dans le volet de navigation, choisissez Hosting, Scripts.
2. Sur la page Scripts, choisissez Créer un script.
3. Sur la page Créer un script, sous Paramètres du script, procédez comme suit :
   a. Dans Nom, entrez un nom de script.
   b. (Facultatif) Pour Version, entrez les informations de version. Comme vous pouvez mettre à jour le contenu d'un script, les données de version peuvent être utiles pour le suivi des mises à jour.
   c. Pour la source du script, choisissez Amazon S3 URI.
   d. Entrez l'URI S3 de l'objet de script que vous avez chargé sur Amazon S3, puis choisissez la version de l'objet. Si vous ne vous souvenez pas de l'URI et de la version de l'objet Amazon S3, choisissez Parcourir S3, puis recherchez l'objet de script.
4. (Facultatif) Sous Balises, ajoutez des balises au script en saisissant des paires clé/valeur.
5. Sélectionnez Create (Créer).

   Amazon GameLift attribue un identifiant au nouveau script et télécharge le fichier .zip désigné. Vous pouvez consulter le nouveau script, y compris son statut, sur la page Scripts.

AWS CLI

Utilisez la `create-script`AWS CLIcommande pour définir le nouveau script et charger les fichiers de script de votre serveur.

1. Ouvrez une fenêtre de ligne de commande et accédez à un répertoire dans lequel vous pouvez utiliser leAWS CLI.
2. Entrez la create-script commande et les paramètres suivants. Le --storage-location paramètre indique l'emplacement du compartiment Amazon S3 de vos fichiers de script.

```
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
```

Example (Exemple)

```
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
```

En réponse à votre demande, Amazon GameLift renvoie le nouvel objet de script.

3. Pour voir le nouveau script, appelez `describe-script`.

Mettre à jour les fichiers de script

Vous pouvez mettre à jour les métadonnées d'une ressource de script à l'aide de la GameLift console Amazon ou de la `update-script`AWS CLIcommande.

Vous pouvez également mettre à jour le contenu du script pour une ressource de script. Amazon GameLift déploie le contenu des scripts sur toutes les instances de flotte qui utilisent la ressource de script mise à jour. Lorsque le script mis à jour est déployé, les instances l'utilisent pour démarrer de

nouvelles sessions de jeu. Les sessions de jeu déjà en cours au moment de la mise à jour n'utilisent pas le script mis à jour.

Pour mettre à jour les fichiers de script

- Pour les fichiers de script stockés localement, pour charger le fichier script .zip mis à jour, utilisez la GameLift console Amazon ou la update-script commande.
- Pour les fichiers de script stockés dans un compartiment Amazon S3, chargez les fichiers de script mis à jour dans le compartiment S3. Amazon vérifie GameLift régulièrement la présence de fichiers de script mis à jour et les récupère directement depuis le compartiment S3.

Configuration des GameLift flottes Amazon

Cette section fournit des informations détaillées sur la conception, la construction et la maintenance de flottes destinées à être utilisées avec AmazonGameLift. Vous pouvez utiliser les GameLift flottes Amazon pour déployer des serveurs de jeu personnalisés et des serveurs en temps réel.

Une flotte représente vos ressources d'hébergement sous la forme d'un ensemble d'instances Amazon Elastic Compute Cloud (Amazon EC2) ou de matériel physique. L'emplacement d'une flotte détermine où les instances ou le matériel sont déployés pour héberger les sessions de jeu de vos joueurs. La taille d'une flotte, ainsi que le nombre de sessions de jeu et de joueurs qu'elle peut prendre en charge, dépendent du nombre d'instances ou de la quantité de matériel que vous lui donnez. Vous pouvez ajuster les instances virtuelles manuellement ou à l'aide du dimensionnement automatique.

De nombreux jeux en production utilisent plusieurs flottes. Vous pouvez utiliser plusieurs flottes, par exemple pour faire fonctionner simultanément plusieurs versions de votre serveur de jeu, pour fournir une capacité de sauvegarde aux flottes Spot ou pour intégrer une redondance.

Pour savoir comment créer des flottes adaptées aux besoins de votre jeu, commencez parGuide GameLift de conception de flotte Amazon. Une fois que votre flotte est opérationnelleÉlargir la capacité GameLift d'hébergement d'Amazon, voirAjouter un alias à une GameLift flotte Amazon, etConfiguration des GameLift files d'attente Amazon pour le placement des sessions de jeu.

Rubriques

- Guide GameLift de conception de flotte Amazon
- Créez une nouvelle GameLift flotte Amazon
- Gérez vos GameLift flottes Amazon

- Ajouter un alias à une GameLift flotte Amazon
- Résoudre les problèmes liés à la GameLift flotte Amazon
- Connectez-vous à distance aux instances GameLift de flotte Amazon

Guide GameLift de conception de flotte Amazon

Ce guide de conception décrit les meilleures pratiques pour créer un parc de ressources d'hébergement à utiliser avec AmazonGameLift. Choisissez une combinaison de ressources d'hébergement et découvrez comment les configurer en fonction de votre jeu.

Rubriques

- Choisir les ressources GameLift informatiques d'Amazon
- Gérez la façon dont Amazon GameLift lance ses serveurs de jeux
- Utiliser des instances Spot avec Amazon GameLift

Choisir les ressources GameLift informatiques d'Amazon

Pour déployer vos serveurs de jeu et héberger des sessions de jeu pour vos joueurs, Amazon GameLift utilise les ressources Amazon Elastic Compute Cloud (Amazon EC2) appelées instances, ou votre matériel physique. Lorsque vous configurez une nouvelle flotte à l'aide d'instances, déterminez le type d'instances dont vous avez besoin et comment exécuter les processus du serveur de jeu sur celles-ci. Lorsqu'une flotte EC2 gérée est active et prête à accueillir des sessions de jeu, vous pouvez ajouter ou supprimer des instances selon les besoins des joueurs.

Vous pouvez déployer vos serveurs de GameLift jeu Amazon sur une combinaison de deux types de calcul :

- EC2 géré : les flottes EC2 gérées utilisent des instances Amazon EC2 pour héberger vos serveurs de jeu. Amazon GameLift gère les instances et élimine le fardeau de la gestion du matériel et des logiciels lié à l'hébergement de vos jeux.
- Amazon GameLift Anywhere — GameLift Anywhere Les flottes Amazon utilisent votre infrastructure existante pour héberger des serveurs de jeux tandis qu'Amazon GameLift gère votre matchmaking et vos files d'attente.

Lorsque vous choisissez les ressources de calcul pour votre flotte, tenez compte des facteurs suivants :

- Matériel disponible
- Emplacement de la flotte
- Instances à la demande et instances ponctuelles
- Operating systems
- Types d'instances
- Quotas de service

Matériel disponible

Tenez compte de l'infrastructure existante lors de votre mise en œuvre. Pendant que vous migrez des jeux vers Amazon GameLift, vous pouvez continuer à utiliser votre infrastructure. Avec Amazon GameLift Anywhere, vous pouvez utiliser votre propre infrastructure ainsi que des instances EC2 GameLift gérées par Amazon. Vous pouvez également utiliser votre infrastructure existante pour héberger des jeux plus près de vos joueurs que ne le permettent GameLift les sites Amazon compatibles. Pour plus d'informations sur la configuration des GameLift Anywhere flottes Amazon, consultezCréez une GameLift Anywhere flotte Amazon.

Emplacement de la flotte

Tenez compte des emplacements géographiques dans lesquels vous prévoyez de déployer vos serveurs de jeu. La disponibilité du type d'instance varie en fonction Région AWS de la zone locale.

Pour les flottes multisites, la disponibilité des instances et les quotas dépendent de la combinaison de la région d'origine de la flotte et des sites distants sélectionnés. Pour plus d'informations sur les emplacements des flottes, consultezSites GameLift d'hébergement Amazon.

Pour les GameLift Anywhere flottes Amazon, vous déterminez l'emplacement de votre matériel physique. Pour plus d'informations sur les emplacements personnalisés, consultezAmazon GameLift Anywhere.

Instances à la demande et instances ponctuelles

Les instances à la demande Amazon EC2 et les instances Spot offrent le même matériel et les mêmes performances, mais leur disponibilité et leur coût diffèrent.

On-Demand instances

Vous pouvez acquérir une instance à la demande lorsque vous en avez besoin et la conserver aussi longtemps que vous le souhaitez. Les instances à la demande ont un coût fixe, ce qui signifie que

vous payez en fonction de la durée pendant laquelle vous les utilisez, et il n'y a aucun engagement à long terme.

Spot instances

Les instances Spot peuvent constituer une alternative rentable aux instances à la demande en utilisant la capacité AWS informatique inutilisée. Les prix des instances Spot fluctuent en fonction de l'offre et de la demande pour chaque type d'instance sur chaque site. AWS peut interrompre les instances Spot chaque fois qu'elle a besoin de récupérer sa capacité. Amazon GameLift utilise les files d'attente et l'algorithme FleetIQ pour déterminer si AWS une instance Spot va être interrompue. Cela met l'instance en état de recyclage. Ensuite, lorsqu'aucune session de jeu n'est active sur l'instance, Amazon GameLift essaie de la remplacer.

Pour plus d'informations sur l'utilisation des instances Spot, consultezUtiliser des instances Spot avec Amazon GameLift.

Operating systems

GameLift Les instances Amazon prennent en charge les versions de serveurs de jeu qui s'exécutent sous Microsoft Windows ou Amazon Linux. Lorsque vous importez une version de jeu sur Amazon GameLift, spécifiez le système d'exploitation du jeu. Lorsque vous créez une flotte Amazon EC2 pour déployer le build du jeu, Amazon configure GameLift automatiquement les instances avec le système d'exploitation du build. Pour plus d'informations sur les systèmes d'exploitation de serveurs de jeu pris en charge, consultezAssistance au développement avec Amazon GameLift.

Lorsque vous utilisez une GameLift Anywhere flotte Amazon, vous pouvez utiliser n'importe quel système d'exploitation compatible avec votre matériel. GameLift AnywhereLes flottes Amazon vous obligent à déployer votre version de jeu sur le matériel tout en utilisant Amazon GameLift pour gérer vos ressources en un seul endroit.

Types d’instances

Le type d'instance d'un parc Amazon EC2 détermine le type de matériel utilisé par les instances. Les différents types d'instances offrent différentes combinaisons de puissance de calcul, de mémoire, de stockage et de capacités réseau.

Lorsque vous choisissez parmi les types d'instances disponibles pour votre jeu, tenez compte des points suivants :

- Architecture de calcul de votre serveur de jeu : x64 ou Arm (AWS Graviton).

> **Note**
>
> Les instances Graviton Arm nécessitent un GameLift serveur Amazon basé sur le système d'exploitation Linux. Le SDK Server 5.1.1 ou une version ultérieure est requis pour C++ et C#. Le SDK Server 5.0 ou une version ultérieure est requis pour Go. Ces instances ne prennent pas out-of-the-box en charge l'installation de Mono sur Amazon Linux 2023 (AL2023) ou Amazon Linux 2 (AL2).

- Les exigences en matière de calcul, de mémoire et de stockage de la version de votre serveur de jeu.
- Le nombre de processus de serveur que vous prévoyez d'exécuter par instance.

En utilisant un type d'instance plus important, vous pourrez peut-être exécuter plusieurs processus de serveur sur chaque instance. Cela peut réduire le nombre d'instances nécessaires pour répondre à la demande des joueurs.

Pour plus d'informations, consultez:

- À propos des types d'instances, consultez la section Types d'instances Amazon EC2.
- À propos de l'exécution de plusieurs processus par instance, voirGérez la façon dont Amazon GameLift lance ses serveurs de jeux.

Quotas de service

Pour connaître les quotas de service par défaut pour Amazon GameLift et les quotas actuels pour le vôtre Compte AWS, procédez comme suit :

- Pour obtenir des informations générales sur les quotas de service pour Amazon GameLift, consultez la section GameLiftPoints de terminaison et quotas Amazon dans le Références générales AWS.
- Pour obtenir la liste des types d'instances disponibles par emplacement pour votre compte, ouvrez la page Quotas de service de la GameLift console Amazon. Cette page affiche également l'utilisation actuelle de votre compte pour chaque type d'instance dans chaque emplacement.
- Pour obtenir la liste des quotas actuels de votre compte pour les types d'instances par région, exécutez la commande AWS Command Line Interface (AWS CLI) `describe-ec2-instance-`

`limits`. Cette commande renvoie le nombre d'instances actives que vous avez dans votre région par défaut (ou dans une autre région que vous spécifiez).

Alors que vous vous préparez à lancer votre jeu, remplissez un questionnaire de lancement sur la GameLift console Amazon. L' GameLift équipe Amazon utilise le questionnaire de lancement pour déterminer les quotas et les limites appropriés pour votre jeu.

Gérez la façon dont Amazon GameLift lance ses serveurs de jeux

Vous pouvez configurer la configuration d'exécution d'un parc EC2 géré pour exécuter plusieurs processus de serveur de jeu par instance. Cela permet d'utiliser vos ressources d'hébergement de manière plus efficace.

Comment une flotte gère plusieurs processus

Amazon GameLift utilise la configuration d'exécution d'un parc pour déterminer le type et le nombre de processus à exécuter sur chaque instance. Une configuration d'exécution contient au moins une configuration de processus serveur qui représente un serveur de jeu exécutable. Vous pouvez définir des configurations de processus de serveur supplémentaires pour exécuter d'autres types de processus liés à votre jeu. Chaque configuration de processus serveur contient les informations suivantes :

- Nom de fichier et chemin d'accès d'un exécutable dans votre build de jeu.
- (Facultatif) Paramètres à transmettre au processus serveur au lancement.
- Le nombre de processus à exécuter simultanément.

Lorsqu'une instance du parc est activée, elle lance l'ensemble des processus serveur définis dans la configuration du runtime. Dans le cas de plusieurs processus, Amazon GameLift échelonne le lancement de chaque processus. Les processus du serveur ont une durée de vie limitée. À leur fin, Amazon GameLift lance de nouveaux processus pour maintenir le nombre et le type de processus serveur définis dans la configuration d'exécution.

Vous pouvez modifier la configuration d'exécution d'une flotte à tout moment en ajoutant, en modifiant ou en supprimant des configurations de processus serveur. Chaque instance vérifie régulièrement si des mises à jour ont été apportées à la configuration d'exécution du parc afin de mettre en œuvre les modifications. Voici comment Amazon GameLift adopte les modifications de configuration de l'environnement d'exécution :

1. L'instance envoie une demande à Amazon GameLift pour obtenir la dernière version de la configuration d'exécution.
2. L'instance compare ses processus actifs à la dernière configuration d'exécution, puis effectue les opérations suivantes :
   - Si la configuration d'exécution mise à jour supprime un type de processus serveur, les processus serveur actifs de ce type continuent de s'exécuter jusqu'à leur fin. L'instance ne remplace pas ces processus de serveur.
   - Si la configuration d'exécution mise à jour réduit le nombre de processus concurrents pour un type de processus serveur, les processus serveur excédentaires de ce type continuent de s'exécuter jusqu'à leur fin. L'instance ne remplace pas ces processus de serveur excédentaires.
   - Si la configuration d'exécution mise à jour ajoute un nouveau type de processus serveur ou augmente le nombre de processus concurrents pour un type existant, l'instance lance de nouveaux processus serveur, jusqu'au GameLift maximum autorisé par Amazon. Dans ce cas, l'instance lance de nouveaux processus serveur à la fin des processus existants.

Optimisez un parc pour de multiples processus

Pour utiliser plusieurs processus sur un parc, procédez comme suit :

- Créez une version contenant les exécutables du serveur de jeu que vous souhaitez déployer sur une flotte, puis importez la version sur Amazon. GameLift Tous les serveurs de jeu d'une version doivent fonctionner sur la même plateforme et utiliser le SDK Amazon GameLift Server.
- Créez une configuration d'exécution avec une ou plusieurs configurations de processus serveur et plusieurs processus simultanés.
- Intégrez les clients de jeu à la version 2016-08-04 ou ultérieure du AWS SDK.

Pour optimiser les performances de votre flotte, nous vous recommandons de procéder comme suit :

- Gérez les scénarios d'arrêt des processus du serveur afin qu'Amazon GameLift puisse recycler les processus de manière efficace. Par exemple :
  - Ajoutez une procédure d'arrêt au code de votre serveur de jeu qui appelle l'API du serveur`ProcessEnding()`.
  - Implémentez la fonction de rappel `OnProcessTerminate()` dans le code de votre serveur de jeu pour gérer les demandes de résiliation d'AmazonGameLift.

- Assurez-vous qu'Amazon GameLift arrête et relance les processus de serveur défectueux. Signalez l'état de santé à Amazon en GameLift implémentant la fonction de `OnHealthCheck()` rappel dans le code de votre serveur de jeu. Amazon arrête GameLift automatiquement les processus du serveur signalés comme défectueux pendant trois rapports consécutifs. Si vous ne l'implémentez pas`OnHealthCheck()`, Amazon GameLift suppose qu'un processus serveur est sain, à moins que le processus ne réponde pas à une communication.

Choisissez le nombre de processus par instance

Lorsque vous décidez du nombre de processus simultanés à exécuter sur une instance, tenez compte des points suivants :

- Amazon GameLift limite chaque instance à un nombre maximum de processus simultanés. La somme de tous les processus simultanés pour les configurations des processus de serveur d'un parc ne peut pas dépasser ce quota.
- Pour maintenir des niveaux de performance acceptables, le type d'instance Amazon EC2 peut limiter le nombre de processus pouvant s'exécuter simultanément. Testez différentes configurations pour votre jeu afin de trouver le bon nombre de processus pour votre type d'instance préféré.
- Amazon GameLift n'exécute pas plus de processus simultanés que le nombre total configuré. Cela signifie que la transition de la configuration d'exécution précédente vers la nouvelle configuration peut se faire progressivement.

Utiliser des instances Spot avec Amazon GameLift

Lorsque vous configurez votre flotte EC2 GameLift gérée par Amazon, vous pouvez utiliser des instances Spot, des instances à la demande ou une combinaison des deux. Découvrez comment Amazon GameLift utilise les instances Spot dansInstances à la demande et instances ponctuelles. Pour utiliser les flottes de spots, l'intégration de votre jeu nécessite les ajustements répertoriés sur cette page.

Utilisez-vous FlexMatch pour le matchmaking ? Vous pouvez ajouter des flottes ponctuelles à vos files d'attente de session de jeu existantes pour les placements de mise en relation.

1. Concevez votre file d'attente de sessions de jeu pour les instances Spot.

La gestion du placement des sessions de jeu par le biais d'une file d'attente est une bonne pratique, et elle est obligatoire lors de l'utilisation d'instances Spot. Pour concevoir votre file d'attente, considérez les points suivants :

- Emplacements — Pour optimiser l'expérience des joueurs, choisissez des emplacements géographiquement proches de vos joueurs.
- Types d'instances — Tenez compte de la configuration matérielle requise pour vos serveurs de jeu et de la disponibilité des instances dans les emplacements que vous avez choisis.

Pour essayer une file d'attente qui optimise la disponibilité et la résilience de Spot, consultez. Tutoriel : Configuration d'une file d'attente de sessions de jeu pour les instances Spot

2. Créez les flottes pour votre file d'attente optimisée pour les instances Spot.

   En fonction de la conception de votre file d'attente, créez des flottes pour déployer vos serveurs de jeu aux emplacements et aux types d'instances souhaités. Consultez Créez une flotte GameLift gérée par Amazon pour plus de détails sur la création et la configuration de nouvelles flottes.

3. Créez votre file d'attente de session de jeu.

   Ajoutez les destinations de la flotte, configurez le processus de placement des sessions de jeu et définissez les priorités de placement. Consultez Création d'une file d'attente de sessions de jeu pour plus de détails sur la création et la configuration de la nouvelle file d'attente.

4. Mettez à jour le service client de votre jeu pour utiliser la file d'attente.

   Lorsque votre client de jeu utilise une file d'attente pour demander des ressources, celle-ci évite les ressources présentant un risque élevé d'interruption et sélectionne l'emplacement qui correspond à vos priorités définies. Pour obtenir de l'aide sur la mise en œuvre des placements de sessions de jeu dans votre client de jeu, consultez Créez des sessions de jeu.

5. Mettez à jour votre serveur de jeu pour qu'il puisse gérer une interruption ponctuelle.

   AWS peut interrompre les instances Spot avec une notification de 2 minutes, lorsqu'elles ont besoin de récupérer leur capacité. Configurez votre serveur de jeu pour gérer les interruptions afin de minimiser l'impact sur les joueurs.

   Avant AWS de récupérer une instance Spot, celle-ci envoie une notification de résiliation. Amazon GameLift transmet la notification à tous les processus du serveur concernés en

invoquant la fonction de rappel GameLift du SDK Amazon Server. `onProcessTerminate()` Implémentez ce rappel pour mettre fin à la session de jeu ou déplacer la session de jeu et les joueurs vers une nouvelle instance. Consultez Répondre à une notification d'arrêt d'un processus serveur pour plus de détails sur l'implémentation de `onProcessTerminate()`.

> **Note**
>
> AWS met tout en œuvre pour fournir la notification avant de récupérer une instance, mais il est possible qu' AWS elle récupère l'instance Spot avant l'arrivée de l'avertissement. Préparez votre serveur de jeu à faire face aux interruptions inattendues.

6. Passez en revue les performances de vos flottes et files d'attente Spot.

   Consultez GameLift les statistiques Amazon dans la GameLift console Amazon ou avec Amazon CloudWatch pour évaluer les performances. Pour plus d'informations sur GameLift les métriques Amazon, consultezSurveillez Amazon GameLift avec Amazon CloudWatch. Les métriques clés incluent :

   - Taux d'interruption : utilisez les `GameSessionInterruptions` indicateurs `InstanceInterruptions` et pour suivre le nombre et la fréquence des interruptions liées à Spot pour les instances et les sessions de jeu. Les sessions de jeu récupérées par AWS ont un statut `TERMINATED` et une raison de statut de`INTERRUPTED`.
   - Efficacité des files d'attente : suivez les taux de réussite du placement, le temps d'attente moyen et la profondeur des files d'attente pour vous assurer que les flottes Spot n'ont aucun impact sur les performances de votre file d'attente.
   - Utilisation de la flotte : surveillez les données relatives aux instances, aux sessions de jeu et aux sessions des joueurs. L'utilisation de vos flottes à la demande peut indiquer que les files d'attente évitent de se placer dans vos flottes Spot afin d'éviter toute interruption.

Créez une nouvelle GameLift flotte Amazon

Créez une nouvelle flotte et déployez votre version de serveur de jeu personnalisée ou vos serveurs en temps réel pour l'hébergement. Vous pouvez déployer n'importe quelle ressource de script ou de build de jeu que vous importez sur AmazonGameLift.

Rubriques

- Comment fonctionne la création GameLift de flottes Amazon

- Créez une flotte GameLift gérée par Amazon
- Créez une GameLift Anywhere flotte Amazon

Comment fonctionne la création GameLift de flottes Amazon

Lorsque vous créez une nouvelle flotte, Amazon GameLift lance un flux de travail qui crée une flotte avec une instance Amazon Elastic Compute Cloud (Amazon EC2) dans chaque emplacement de flotte. Au fur et à mesure qu'Amazon GameLift termine chaque étape du flux de travail, la flotte émet des événements et Amazon GameLift met à jour l'état de la flotte. Vous pouvez suivre tous les événements à l'aide de la GameLift console Amazon ou en appelant l'opération GameLift d'API Amazon DescribeFleetEvents. Vous pouvez également suivre l'état de chaque site à l'aide de DescribeFleetLocationAttributes.

Flux de travail de création de flottes EC2 :

- Amazon GameLift crée une ressource de flotte dans la région d'origine de la flotte et dans chaque emplacement distant défini dans la flotte.
- Amazon GameLift définit la capacité souhaitée sur une instance.
- Amazon GameLift définit le statut de la flotte et de l'emplacement sur Nouveau.
- Amazon GameLift commence à enregistrer les événements dans le journal des événements de la flotte.
- Amazon GameLift alloue les ressources informatiques demandées pour une nouvelle instance dans chaque emplacement du parc.
- Amazon GameLift télécharge les fichiers du serveur de jeu sur chaque instance et définit l'état du parc sur Téléchargement.
- Amazon GameLift valide les fichiers du serveur de jeu téléchargés sur chaque instance afin de vérifier qu'aucune erreur ne s'est produite lors du téléchargement. Amazon GameLift définit le statut de la flotte sur Validation.
- Amazon GameLift crée le serveur de jeu sur chaque instance et définit le statut de la flotte sur Building.
- Amazon GameLift commence à lancer des processus de serveur sur chaque instance, en suivant les instructions de la configuration d'exécution du parc. Si vous avez configuré le parc pour exécuter plusieurs processus de serveur simultanés par instance, Amazon GameLift étale le lancement du processus de quelques secondes. Au fur et à mesure que chaque processus est

mis en ligne, Amazon est informé de son état de préparationGameLift. Amazon GameLift définit le statut de la flotte sur Activation.

- Amazon GameLift définit les statuts du parc et les statuts de localisation sur Actif lorsque les processus du serveur signalent qu'ils sont prêts.

Amazon GameLift Anywhere fleet creation

- Amazon GameLift crée une ressource de flotte. Pour la région d'origine du parc et chaque emplacement personnalisé défini dans le parc, Amazon GameLift définit le statut du parc et de l'emplacement sur Nouveau.
- Amazon GameLift commence à enregistrer les événements dans le journal des événements de la flotte.
- Une fois qu'un processus de serveur d'un parc indique à Amazon GameLift qu'il est prêt, Amazon GameLift définit l'état du parc et le statut de localisation sur Actif. Lorsque les processus de serveur d'autres sites de flotte indiquent qu'ils sont prêts, Amazon GameLift définit le statut de chaque emplacement de flotte sur Actif.

Pour obtenir de l'aide sur la résolution des problèmes liés à la création de flottes, consultezRésoudre les problèmes liés à la GameLift flotte Amazon.

Créez une flotte GameLift gérée par Amazon

Utilisez la GameLift console Amazon ou le AWS Command Line Interface (AWS CLI) pour créer une flotte gérée.

Une fois que vous avez créé une nouvelle flotte EC2 gérée, le statut de la flotte passe par plusieurs étapes au fur et à mesure qu'Amazon GameLift déploie la flotte, installe et démarre les serveurs de jeu. La flotte est prête à accueillir des sessions de jeu une fois qu'elle aura atteint `ACTIVE` son statut. Pour obtenir de l'aide face à des problèmes de création de flotte, veuillez consulter Résoudre les problèmes liés à la GameLift flotte Amazon.

Console

Pour créer une flotte EC2 gérée

1. Dans la GameLift console Amazon, dans le volet de navigation, choisissez Fleets.
2. Sur la page Fleets (Flottes), choisissez Create fleet (Créer une flotte).

3. Choisissez Managed EC2.
4. Sur la page des détails de la flotte, procédez comme suit :

   a. Dans Nom, entrez un nom de flotte. Nous vous recommandons d'inclure le type de flotte (sur place ou à la demande) dans le nom de votre flotte. Il est ainsi beaucoup plus facile d'identifier les types de flottes lorsque vous consultez une liste de flottes.

   b. Dans Description, fournissez une brève description de la flotte.

   c. Pour le type binaire, sélectionnez Build ou Script pour définir le type de serveur de jeu qu'Amazon GameLift déploie sur cette flotte.

   d. Sélectionnez un script ou un build dans la liste déroulante des scripts ou des builds téléchargés.
5. (Facultatif) Sous Détails supplémentaires pour les éléments suivants :

   a. Pour Rôle d'instance, spécifiez un rôle IAM qui autorise les applications de votre version de jeu à accéder aux autres AWS ressources de votre compte. Pour en savoir plus, consultez Communiquez avec les autres AWS ressources de vos flottes. Pour créer une flotte avec un rôle d'instance, votre compte doit disposer de l'`PassRole`autorisation IAM. Pour en savoir plus, consultez Exemples d'autorisations IAM pour Amazon GameLift.

   Si vous souhaitez autoriser des applications qui ne sont pas des exécutables sur le serveur, telles qu'un CloudWatch agent, activez l'option d'informations d'identification partagées.

   Vous ne pouvez pas mettre à jour ces paramètres après la création de la flotte.

   b. Pour la génération de certifications, choisissez de demander à Amazon de GameLift générer un certificat TLS pour le parc. Vous pouvez utiliser un certificat TLS de flotte pour que votre client de jeu authentifie un serveur de jeux lors de la connexion, et chiffrer toutes les communications client/serveur. Pour chaque instance d'un parc compatible TLS, Amazon crée GameLift également une nouvelle entrée DNS avec le certificat. Utilisez ces ressources pour configurer l'authentification et le chiffrement pour votre jeu.

   c. Pour Groupe de mesures, entrez le nom d'un groupe de mesures de flotte nouveau ou existant. Vous pouvez agréger les mesures de plusieurs flottes en les ajoutant au même groupe de mesures.

   Vous ne pouvez pas mettre à jour le groupe de mesures après la création de la flotte.
6. Choisissez Suivant.

7. Sur la page Sélectionner des emplacements, sélectionnez un ou plusieurs sites distants supplémentaires sur lesquels déployer des instances. La région d'accueil est automatiquement sélectionnée en fonction de la région à partir de laquelle vous accédez à la console. Si vous sélectionnez des sites supplémentaires, des instances de flotte sont également déployées sur ces sites.

   > **Important**
   >
   > Pour utiliser des régions qui ne sont pas activées par défaut, activez-les dans votreCompte AWS.
   >
   > - Les flottes dont les régions ne sont pas activées et que vous avez créées avant le 28 février 2022 ne sont pas affectées.
   > - Pour créer de nouvelles flottes multi-sites ou pour mettre à jour les flottes multi-sites existantes, activez d'abord les régions que vous choisissez d'utiliser.
   >
   > Pour plus d'informations sur les régions qui ne sont pas activées par défaut et sur la manière de les activer, consultez la section Gestion Régions AWS dans le Références générales AWS.

8. Choisissez Suivant.
9. Sur la page Définir les détails de l'instance, sélectionnez

   a. Instances à la demande ou ponctuelles pour cette flotte. Pour plus d'informations sur les types de flottes, consultezInstances à la demande et instances ponctuelles.

   b. Dans le menu Architecture du filtre, choisissez x64 ou Arm.

   > **Note**
   >
   > Les instances Graviton Arm nécessitent un GameLift serveur Amazon basé sur le système d'exploitation Linux. Le SDK Server 5.1.1 ou une version ultérieure est requis pour C++ et C#. Le SDK Server 5.0 ou une version ultérieure est requis pour Go. Ces instances ne prennent pas out-of-the-box en charge l'installation de Mono sur Amazon Linux 2023 (AL2023) ou Amazon Linux 2 (AL2).

Pour plus d'informations sur les architectures Amazon EC2 Arm, consultez les sections Processor AWS Graviton et types d'instances Amazon EC2.

Pour plus d'informations sur les types d'instances pris en charge par Amazon GameLift, consultez les `EC2InstanceType` valeurs sous les paramètres de demande CreateFleet ().

10. Sélectionnez un type d'instance Amazon EC2 dans la liste. Pour plus d'informations sur le choix d'un type d'instance, consultezTypes d'instances. Après avoir créé le parc, vous ne pouvez pas modifier le type d'instance.
11. Choisissez Suivant.
12. Sur la page Configurer le runtime, sous Configuration du runtime, procédez comme suit :

    a. Dans le champ Launch path, saisissez le chemin d'accès à l'exécutable du jeu dans votre build ou votre script. Sur les instances Windows, les serveurs de jeux sont générés dans le chemin `C:\game`. Sur les instances Linux, les serveurs de jeu sont conçus pour`/local/game`. Exemples : **`C:\game\MyGame\server.exe`**`/local/game/MyGame/server.exe`, ou**`MyRealtimeLaunchScript.js`**.

    b. (Facultatif) Dans Paramètres de lancement, entrez les informations à transmettre à l'exécutable de votre jeu sous forme d'ensemble de paramètres de ligne de commande. Exemple: **`+sv_port 33435 +start_lobby`**.

    c. Pour les processus simultanés, choisissez le nombre de processus serveur à exécuter simultanément sur chaque instance du parc. Consultez les GameLift limites d'Amazon relatives au nombre de processus serveur simultanés.

    Les limites sur les processus serveur simultanés par instance s'appliquent au nombre total de processus simultanés pour toutes les configurations. Si vous configurez le parc de manière à dépasser la limite, le parc ne pourra pas être activé.
13. Sous Activation de session de jeu, indiquez les limites d'activation de nouvelles sessions de jeu sur les instances de ce parc :

    a. Pour le nombre maximal de sessions de jeu simultanées, entrez le nombre de sessions de jeu sur une instance qui s'activent simultanément. Cette limite est utile lorsque le lancement de plusieurs nouvelles sessions de jeu peuvent avoir un impact sur les performances des autres sessions de jeu s'exécutant sur l'instance.

b. Pour Nouveau délai d'activation, entrez le délai d'attente avant l'activation d'une session. Si la session de jeu ne passe pas au `ACTIVE` statut avant l'expiration du délai imparti, Amazon GameLift met fin à l'activation de la session de jeu.

14. (Facultatif) Dans les paramètres du port EC2, procédez comme suit :

    a. Choisissez Ajouter un paramètre de port pour définir les autorisations d'accès pour le trafic entrant se connectant au processus serveur déployé sur le parc.

    b. Pour Type, choisissez TCP personnalisé ou UDP personnalisé.

    c. Pour Plage de ports, entrez une plage de numéros de port autorisant les connexions entrantes. Une plage de ports doit utiliser le format`nnnnn[-nnnnn]`, avec des valeurs comprises entre 1026 et 60 000. Exemple : **`1500`** ou **`1500-20000`**.

    d. Pour la plage d'adresses IP, entrez une plage d'adresses IP. Utilisez la notation CIDR. Exemple : **`0.0.0.0/0`** (Cet exemple autorise l'accès à quiconque essaie de se connecter.)

15. (Facultatif) Sous Paramètres des ressources de session de jeu, procédez comme suit :

    a. Pour la politique de protection contre le dimensionnement du jeu, activez ou désactivez la protection contre le dimensionnement. Amazon GameLift ne mettra pas fin à l'instance avec protection lors d'un événement de réduction si l'entreprise héberge une session de jeu active.

    b. Pour la limite de création de ressources, entrez le nombre maximum de sessions de jeu qu'un joueur peut créer pendant la période de garantie.

16. Choisissez Suivant.

17. (Facultatif) Ajoutez des balises au build en saisissant des paires clé et valeur. Choisissez Suivant pour passer à l'examen de la création de flotte.

18. Sélectionnez Create (Créer). Amazon GameLift attribue un identifiant à la nouvelle flotte et lance le processus d'activation de la flotte. Vous pouvez suivre le statut de la nouvelle flotte sur la page Flottes.

Vous pouvez mettre à jour les métadonnées et la configuration de la flotte à tout moment, quel que soit l'état de la flotte. Pour en savoir plus, consultez Gérez vos GameLift flottes Amazon. Vous pouvez mettre à jour la capacité de la flotte une fois que celle-ci a atteint le statut ACTIF. Pour en savoir plus, consultez Élargir la capacité GameLift d'hébergement d'Amazon. Vous pouvez également ajouter ou supprimer des sites distants.

AWS CLI

Pour créer une flotte avec leAWS CLI, ouvrez une fenêtre de ligne de commande et utilisez la `create-fleet` commande. Pour plus d'informations sur la commande `create-fleet`, consultez `create-fleet` dans la Référence de commande de l'AWS CLI .

L'exemple `create-fleet` ci-dessous crée une nouvelle flotte avec les caractéristiques suivantes :

- La flotte utilise des instances à la demande c5.large avec le système d'exploitation adapté à la version de jeu sélectionnée.
- Il déploie la version de serveur de jeu spécifiée, qui doit être prête, aux emplacements suivants :
  - us-west-2 (région d'origine)
  - sa-east-1 (site distant)
- La génération de certificats TLS est activée.
- Chaque instance de la flotte exécutera dix processus identiques du serveur de jeux simultanément, en activant chaque instance pour héberger jusqu'à dix sessions de jeu simultanément.
- Sur chaque instance, Amazon GameLift autorise l'activation simultanée de deux nouvelles sessions de jeu. Cela met également fin à toute session de jeu en cours d'activation s'ils ne sont pas prêts à héberger des joueurs dans les 300 secondes.
- La protection de session de jeu sera activée pour toutes les sessions de jeu hébergées sur les instances de cette flotte.
- les joueurs individuels peuvent créer trois nouvelles sessions de jeu au cours d'une période de 15 minutes.
- Chaque session de jeu hébergée sur cette flotte possède un point de connexion qui se situe dans les plages d'adresses IP et de ports spécifiées.
- Amazon GameLift ajoute des métriques pour cette flotte au groupe de `EMEAfleets` métriques, qui (dans cet exemple) combine les métriques pour toutes les flottes des régions 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"
```

Si la demande de création de flotte aboutit, Amazon GameLift renvoie un ensemble d'attributs de flotte comprenant les paramètres de configuration que vous avez demandés et un nouvel identifiant de flotte. Amazon lance GameLift ensuite le processus d'activation de la flotte et définit le statut de la flotte et les statuts de localisation sur Nouveau. Vous pouvez suivre le statut de la flotte et consulter d'autres d'informations de flotte à l'aide de ces commandes d'interface de ligne de commande :

- 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

Vous pouvez changer la capacité de la flotte et d'autres paramètres de configuration, si nécessaire, à l'aide des commandes suivantes :

- update-fleet-attributes

- update-fleet-capacity
- update-fleet-port-settings
- update-runtime-configuration
- create-fleet-locations
- delete-fleet-locations

Créez une GameLift Anywhere flotte Amazon

Utilisez Amazon GameLift pour intégrer le matériel de votre environnement à votre hébergement de GameLift jeux Amazon. Amazon GameLift Anywhere enregistre votre matériel auprès d'Amazon GameLift dans un Anywhere parc. Vous pouvez intégrer Anywhere et gérer des flottes EC2 dans les files d'attente des matchmaking et des sessions de jeu pour gérer le matchmaking et le placement des jeux.

Pour plus d'informations sur le test de vos serveurs de jeu avec Amazon GameLift Anywhere, consultezTestez votre intégration à l'aide des GameLift Anywhere flottes Amazon.

Pour commencer, utilisez Assistance au développement avec Amazon GameLift la version 5 ou supérieure et passez en revue les concepts suivants relatifs à l'utilisation d'une GameLift Anywhere flotte Amazon.

Emplacements personnalisés

GameLift AnywhereLes flottes Amazon utilisent des emplacements personnalisés pour représenter les emplacements physiques de votre infrastructure.

Enregistrement de l'appareil

Pour qu'une GameLift Anywhere flotte Amazon puisse communiquer avec vos ressources informatiques, enregistrez d'abord votre appareil. Vous pouvez terminer l'enregistrement de l'appareil depuis le GameLift AWS SDK Amazon en utilisant cette `RegisterCompute`opération. Cette opération utilise l'adresse IP de l'appareil pour l'associer à l'emplacement d'une flotte et pour communiquer avec Amazon GameLift.

Jetons d'authentification

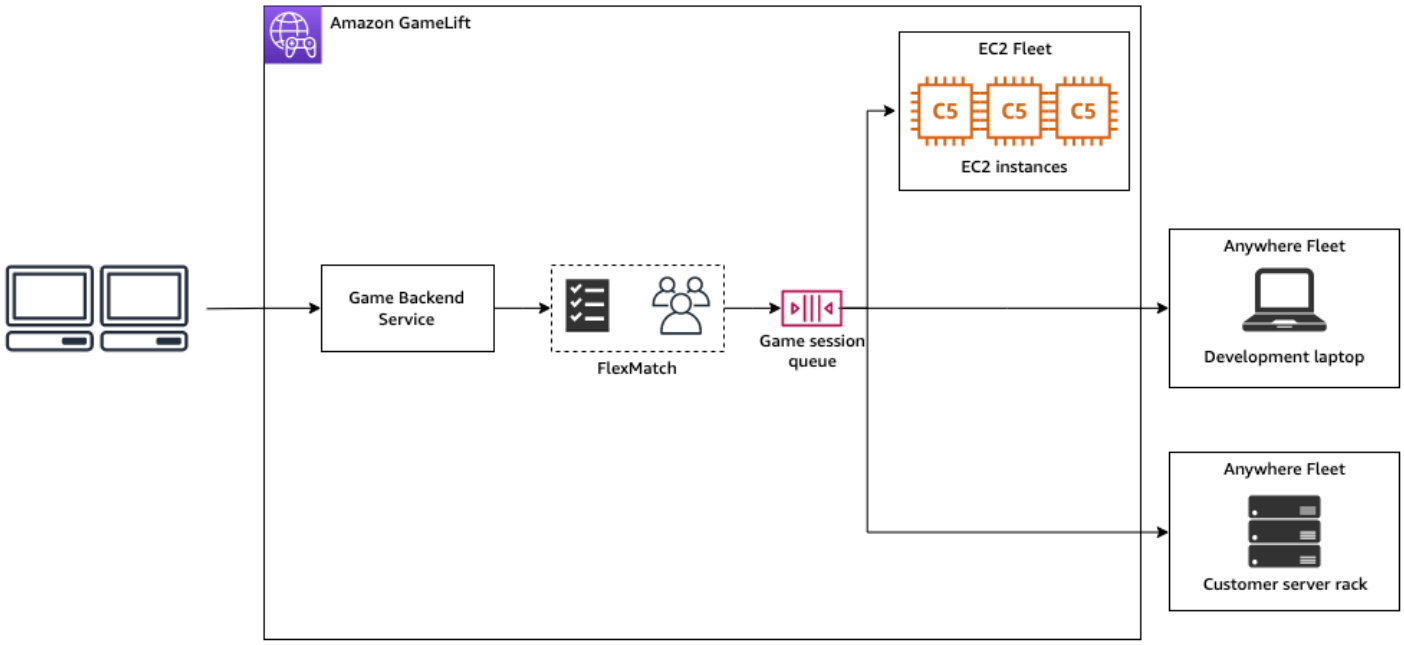
Lorsque vous initialisez un serveur de jeu sur votre ordinateur, le SDK Amazon GameLift Server utilise un jeton d'authentification pour authentifier votre serveur de jeu auprès d'Amazon. GameLift Vous pouvez réutiliser le même jeton d'authentification pour tous les serveurs de jeu sur le

même ordinateur, jusqu'à la date d'expiration du jeton d'authentification. Pour récupérer le jeton d'authentification, appelez la commande `get-compute-auth-token` AWS Command Line Interface (AWS CLI). Passez le jeton à chaque serveur de jeu selon vos besoins.

Sessions de jeu

Chaque session de jeu sur un ordinateur utilise le même jeton d'authentification créé lors de l'enregistrement du calcul sur l'emplacement d'une flotte.

Le schéma suivant montre une file d'attente de session de jeu qui utilise le FlexMatch matchmaking et plusieurs flottes. Les flottes comprennent une flotte EC2 avec des instances C5, une Anywhere flotte avec un ordinateur portable de développement et une flotte avec un rack de serveurs Anywhere hébergé par le client.



Rubriques

- Création d'un emplacement personnalisé
- Création d'une flotte
- Enregistrez votre ordinateur
- Exécuter un processus serveur
- Créez des sessions de jeu
- Migrer vers un EC2 géré

Création d'un emplacement personnalisé

Pour commencer à héberger des jeux sur vos ressources informatiques, créez un emplacement personnalisé décrivant l'emplacement de votre ordinateur.

Console

Pour créer un emplacement personnalisé

1. Ouvrez la GameLift console Amazon.
2. Dans le volet de navigation, sous Hébergement, sélectionnez Locations.
3. Sur la page Emplacements, sélectionnez Créer un emplacement.
4. Dans la boîte de dialogue Créer un emplacement, procédez comme suit :
   a. Entrez un nom de lieu. Cela indique l'emplacement de votre matériel qu'Amazon GameLift utilise pour exécuter vos jeux dans des Anywhere flottes. Amazon GameLift ajoute le nom de votre emplacement personnalisé avec custom-.
   b. (Facultatif) Ajoutez des balises sous forme de paires clé-valeur à votre emplacement personnalisé. Choisissez Ajouter un nouveau tag pour chaque tag que vous souhaitez ajouter.
   c. Choisissez Créer.

AWS CLI

Créez un emplacement personnalisé à l'aide de la `create-location`commande. Les `location-name` étiquettes indiquent l'emplacement de votre matériel qu'Amazon GameLift utilise pour exécuter vos jeux dans des Anywhere flottes. Lorsque vous créez votre position personnalisée, le nom de la position doit commencer par`custom-`.

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

Sortie

```
{
    "Location": {
        "LocationName": "custom-location-1",
        "LocationArn": "arn:aws:gamelift:us-east-1:111122223333:location/custom-
location-1"
```

```
    }
}
```

Création d'une flotte

Utilisez la GameLift console Amazon ou le AWS CLI pour créer une Anywhere flotte.

Une fois que vous avez créé une nouvelle Anywhere flotte, le statut de la flotte passe de `NEW` à`ACTIVE`. Lorsqu'elle atteint son `ACTIVE` statut, la flotte est prête à accueillir des sessions de jeu. Pour obtenir de l'aide face à des problèmes de création de flotte, veuillez consulter Résoudre les problèmes liés à la GameLift flotte Amazon.

Console

Pour créer une Anywhere flotte

1. Ouvrez la GameLift console Amazon.
2. Dans le volet de navigation, sous Hébergement, sélectionnez Fleets.
3. Sur la page Fleets (Flottes), choisissez Create fleet (Créer une flotte).
4. À l'étape Type de calcul, choisissez Anywhere, puis cliquez sur Suivant.
5. À l'étape Détails de la flotte, définissez les détails, puis choisissez Suivant.
6. À l'étape Emplacements personnalisés, sélectionnez l'emplacement personnalisé que vous avez créé, puis cliquez sur Suivant. Amazon sélectionne GameLift automatiquement le domicile Région AWS comme région dans laquelle vous créez le parc. Vous pouvez utiliser la région d'origine pour accéder à vos ressources et les utiliser.
7. Effectuez les étapes de création de flotte restantes, puis choisissez Soumettre pour créer votre Anywhere flotte.

AWS CLI

Créez une Anywhere flotte à l'aide de la `create-fleet`commande. Incluez votre position personnalisée dans`locations`. Amazon GameLift crée la flotte dans votre région d'origine et dans les emplacements personnalisés que vous fournissez. Dans l'exemple suivant, remplacez *`FleetName`* et *`custom-location-1`* par vos propres informations. La variable `custom-location-1` est le nom de l'emplacement créé lors de l'Création d'un emplacement personnaliséétape.

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

Exemple de sortie

```
{
    "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"
    }
}
```

Enregistrez votre ordinateur

Pour enregistrer votre ressource de calcul dans le parc que vous avez créé, utilisez la `register-compute`commande. Remplacez le *`fleet-id`* par le `fleet-id` renvoyé à l'étape précédente ou par l'ARN du parc qui se trouve sur la page de détails de votre flotte dans la console. Remplacez *`compute-name`* le et *`ip-address`* par l'adresse IP de votre ressource de calcul.

> **Note**
>
> Nous vous recommandons d'appeler à la fois `get-compute-auth-token` les commandes `register-compute` et depuis un script ou un gestionnaire de processus distinct de votre serveur de jeu.

```
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
```

Exemple de sortie

```
{
    "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"
    }
}
```

Exécuter un processus serveur

1. Obtenez le jeton d'authentification pour votre ressource de calcul auprès du parc que vous avez créé.

   Votre serveur de jeu utilise le jeton d'authentification pour s'authentifier auprès d'Amazon GameLift. Chaque jeton d'authentification possède une date d'expiration. Pour continuer à utiliser la ressource informatique pour héberger votre serveur de jeu, récupérez un nouveau jeton d'authentification avant son expiration.

   > **Note**
   >
   > Amazon GameLift recommande d'appeler à la fois `get-compute-auth-token` les commandes `register-compute` et depuis un script ou un gestionnaire de processus distinct de votre serveur de jeu.

Dans l'exemple suivant, remplacez le `fleet-id` par l'ARN ou l'ID de flotte de la flotte créée lors des étapes précédentes. Remplacez le `compute-name` par le nom du calcul que vous avez créé à l'aide de la `register-compute` commande lors d'une étape précédente.

```
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
```

Exemple de sortie :

```
{
    "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. Exécutez une instance de l'exécutable de votre serveur de jeu.

Pour exécuter votre serveur de jeu, initialisez-le en appelant `InitSDK()` et en lui transmettant les paramètres de votre serveur. Pour plus d'informations, consultez ServerParameters.

Entrée du SDK du serveur :

```
//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. Une fois que le processus serveur est prêt à héberger une session de jeu, appelez Amazon `ProcessReady()` depuis votre serveur de jeu GameLift. Pour plus d'informations sur les paramètres du processus, voir 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);
```

Créez des sessions de jeu

1. Ajoutez de la logique à votre serveur de jeu afin que le processus de votre serveur réponde au `onStartGameSession()` message par`ActivateGameSession()`. Cette opération n'a aucun paramètre, mais elle envoie un accusé de réception à Amazon indiquant GameLift que votre serveur a reçu et accepté le message de création de session de jeu.

```
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. Depuis le service principal de votre client de jeu, démarrez votre session de jeu à l'aide de la `create-game-session`commande `start-matchmakingstart-game-session-placement`, ou.

```
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
```

Exemple de sortie :

```
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 envoie un onStartGameSession() message à votre processus de serveur enregistré. Le message contient l'GameSessionobjet de l'étape précédente avec les propriétés du jeu, les données des sessions de jeu, les données du système de matchmaking et plus encore sur la session de jeu.

3. Lorsque la session de jeu est terminée, mettez fin au processus du serveur de jeu.

   Entrée du SDK du serveur :

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

4. Lancez un autre processus de serveur de jeu en appelantProcessReady(processParams).

Migrer vers un EC2 géré

Une fois que vous avez développé votre serveur de jeu et que vous êtes prêt à préparer la production, vous pouvez demander à Amazon de GameLift gérer votre matériel. Pour migrer vers un parc EC2 géré, téléchargez votre build sur Amazon GameLift et créez un parc EC2 géré. Pour plus d'informations sur le téléchargement de votre build et la configuration d'une flotte, consultez Téléchargez une version de serveur personnalisée sur Amazon GameLift etCréez une flotte GameLift gérée par Amazon.

Gérez vos GameLift flottes Amazon

Utilisez la GameLift console Amazon ou l'AWSinterface de ligne de commande pour mettre à jour les paramètres de votre flotte, modifier des sites distants ou supprimer une flotte.

Mettre à jour la configuration d'une flotte

Vous pouvez mettre à jour les attributs de flotte mutables, les paramètres de port et les configurations d'exécution à l'aide de la GameLift console Amazon ou de l'AWSinterface de ligne de commande. Pour modifier les limites de mise à l'échelle, reportez-vous à la sectionAdaptez automatiquement la capacité de votre flotte avec Amazon GameLift.

Amazon GameLift console

1. Dans la GameLiftconsole Amazon, dans le volet de navigation, choisissez Fleets.
2. Choisissez le parc que vous souhaitez mettre à jour. Un parc doit être en `ACTIVE` état pour que vous puissiez le modifier.
3. Sur la page détaillée de la flotte, dans l'une des sections suivantes, choisissez Modifier.
   - Paramètres de la flotte
     - Modifiez les attributs de la flotte tels que Name et Description.
     - Ajoutez ou supprimez des groupes de mesures, qu'Amazon CloudWatch utilise pour suivre les GameLift métriques Amazon agrégées pour plusieurs flottes.
     - Mettez à jour les paramètres de limite de création de ressources.
     - Activez ou désactivez la protection des sessions de jeu.
   - Configuration de l'exécution : vous pouvez modifier n'importe lequel des paramètres suivants de vos configurations d'exécution et ajouter ou supprimer des configurations d'exécution.

- Modifiez le chemin de lancement de votre serveur de jeu.
- Ajoutez, supprimez ou modifiez des paramètres de lancement facultatifs.
- Modifiez le nombre de processus simultanés exécutés par vos serveurs de jeu.

- Activation des sessions de jeu : modifiez la façon dont vous souhaitez que les processus du serveur s'exécutent et hébergent les sessions de jeu en mettant à jour le nombre maximum d'activations simultanées de sessions de jeu et le délai d'expiration du nouveau délai d'activation.
- Paramètres du port EC2 : mettez à jour les adresses IP et les plages de ports qui autorisent l'accès entrant à la flotte.

4. Choisissez Confirmer pour enregistrer les modifications.

AWS CLI

Utilisez les commandes AWS CLI suivantes pour mettre à jour un parc :

- update-fleet-attributes
- update-fleet-port-settings
- update-runtime-configuration

Mettre à jour l'emplacement du parc

Vous pouvez ajouter ou supprimer les sites distants d'une flotte à l'aide de la GameLift console Amazon ou de l'AWSinterface de ligne de commande. Vous ne pouvez pas modifier la région d'origine d'une flotte.

Amazon GameLift console

1. Dans la GameLiftconsole Amazon, dans le volet de navigation, choisissez Fleets.
2. Choisissez le parc que vous souhaitez mettre à jour. Un parc doit être en `ACTIVE` état pour que vous puissiez le modifier.
3. Sur la page détaillée de la flotte, choisissez l'onglet Emplacements pour afficher les emplacements de la flotte.

4. Pour ajouter de nouveaux emplacements distants, choisissez Ajouter et sélectionnez les emplacements sur lesquels vous souhaitez déployer des instances. Cette liste n'inclut pas les instances pour lesquelles le type d'instance du parc n'est pas disponible.
5. Une fois les nouveaux emplacements sélectionnés, choisissez Ajouter. Amazon GameLift ajoute les nouveaux emplacements à la liste, avec un statut défini sur`NEW`. Amazon commence GameLift ensuite à approvisionner une instance dans chaque emplacement ajouté et à la préparer pour héberger des sessions de jeu.
6. Pour supprimer des sites distants existants de la flotte, utilisez les cases à cocher pour sélectionner un ou plusieurs sites répertoriés.
7. Une ou plusieurs flottes étant sélectionnées, choisissez Supprimer. Les emplacements supprimés restent dans la liste, avec un statut défini sur`DELETING`. Amazon entame GameLift alors le processus d'arrêt de l'activité dans l'emplacement supprimé. S'il existe des instances actives hébergeant des sessions de jeu, Amazon GameLift utilise le processus de fermeture du serveur de jeu pour mettre fin aux sessions de jeu, fermer les serveurs de jeu et fermer les instances de manière progressive.

AWS CLI

Utilisez les commandes de l'AWSinterface de ligne de commande suivantes pour mettre à jour l'emplacement du parc :

- create-fleet-locations
- delete-fleet-locations

Supprimer une flotte

Vous pouvez supprimer une flotte lorsque vous n'en avez plus besoin. La suppression d'une flotte supprime définitivement toutes les données associées aux sessions de jeu et aux sessions des joueurs, ainsi que les données statistiques collectées. Comme alternative, vous pouvez conserver la flotte, désactiver la scalabilité automatique et mettre à l'échelle manuellement la flotte à 0 instance.

> **Note**
>
> Si la flotte dispose d'une connexion d'appairage VPC, demandez d'abord l'autorisation en appelant. CreateVpcPeeringAuthorization Amazon GameLift supprime la connexion d'appairage VPC lors de la suppression du parc.

Vous pouvez utiliser la GameLift console Amazon ou l'outil AWS CLI pour supprimer une flotte.

Amazon GameLift console

1. Dans la GameLiftconsole Amazon, dans le volet de navigation, choisissez Fleets.
2. Choisissez la flotte que vous souhaitez supprimer. Vous pouvez uniquement supprimer des flottes dans `ACTIVE` ou dans `ERROR` leur statut.
3. Choisissez Delete (Supprimer).
4. Dans la boîte de dialogue Supprimer le parc, confirmez la suppression en saisissant**`delete`**.
5. Choisissez Delete (Supprimer).

AWS CLI

Utilisez la commande AWS CLI suivante pour supprimer un parc :

- delete-fleet

Ajouter un alias à une GameLift flotte Amazon

Un GameLift alias Amazon est utilisé pour extraire une désignation de flotte. Les désignations des flottes indiquent à Amazon GameLift où rechercher les ressources disponibles lors de la création de nouvelles sessions de jeu pour les joueurs. Utilisez des alias plutôt que des identifiants de flotte spécifiques pour transférer facilement le trafic des joueurs d'une flotte à l'autre en modifiant l'emplacement cible de l'alias.

Il existe deux types de stratégies de routage pour les alias :

- Simple : achemine le trafic des joueurs vers un numéro de flotte spécifique. Vous pouvez mettre à jour l'ID de flotte d'un alias à tout moment.

- Terminal : renvoie un message au client. Par exemple, vous pouvez diriger les joueurs qui utilisent un out-of-date client vers un endroit où ils peuvent obtenir une mise à niveau.

Les flottes ont une durée de vie limitée, et il existe plusieurs raisons de changer de flotte pendant la durée de vie d'un jeu. Vous ne pouvez pas mettre à jour la version du serveur de jeu d'une flotte ni modifier certains attributs des ressources informatiques d'une flotte existante. Créez plutôt de nouvelles flottes avec les modifications, puis transférez les joueurs vers les nouvelles flottes. Avec des alias, le changement de flottes a un impact minimal sur votre jeu et est invisible pour les joueurs.

Les alias sont utiles dans les jeux qui n'utilisent pas de files d'attente. Pour changer de flottes dans une file d'attente, il suffit de créer une nouvelle, de l'ajouter à la file d'attente et de supprimer l'ancienne flotte, aucune de ces opérations n'étant visible pour les joueurs. En revanche, les clients du jeu qui n'utilisent pas les files d'attente doivent spécifier la flotte à utiliser lorsqu'ils communiquent avec le GameLift service Amazon. Sans alias, un changement de flotte nécessite des mises à jour de votre code de jeu et éventuellement la distribution de clients de jeu mis à jour aux joueurs.

Lors de la mise à jour de l'identifiant de flotte vers lequel pointe un alias, il y a une période de transition de 2 minutes maximum au cours de laquelle les sessions de jeu sur l'alias peuvent se terminer sur l'ancienne flotte.

Créez un nouvel alias

Vous pouvez créer un alias à l'aide de la GameLift console Amazon, comme décrit ici, ou à l'aide de la commande de la AWS CLI create-alias.

1. Dans le volet de navigation de la GameLift console Amazon, sélectionnez Aliases.
2. Dans l'onglet Alias, choisissez Créer un alias. Nous vous recommandons d'inclure le type de flotte dans vos alias. Il est ainsi beaucoup plus facile d'identifier le type de flotte lors de l'affichage d'une liste d'alias.
3. Sur la page Créer un alias, sous Détails de l'alias, procédez comme suit :
   a. Dans Nom, entrez un nom d'alias.
   b. Pour la description, entrez une brève description à des fins d'identification.
   c. Choisissez le type de routage simple ou terminal.
4. (Facultatif) Sous Balises, ajoutez des balises à l'alias en saisissant des paires clé et valeur.
5. Choisissez Créer.

Modifier un alias

Vous pouvez modifier un alias à l'aide de la GameLift console Amazon ou de la commande update-alias de la AWS CLI.

1. Dans le volet de navigation de la GameLift console Amazon, sélectionnez Aliases.
2. Sur la page Alias, choisissez l'alias que vous souhaitez modifier.
3. Sur la page d'alias, choisissez Modifier.
4. Sur la page Éditer un alias, vous pouvez modifier les éléments suivants :
   - Nom d'alias : nom convivial pour votre alias.
   - Description — Brève description de votre alias.
   - Type — Stratégie de routage pour le trafic des joueurs. Sélectionnez Simple pour changer la flotte associée ou sélectionnez Terminal pour modifier le message de fin.
5. Sélectionnez Enregistrer les modifications.

Résoudre les problèmes liés à la GameLift flotte Amazon

Cette rubrique fournit des conseils sur les problèmes de configuration du parc pour une solution d'hébergement GameLift géré Amazon. Pour un dépannage supplémentaire, vous pouvez accéder à distance à une instance de flotte une fois que la flotte est active. Consultez Connectez-vous à distance aux instances GameLift de flotte Amazon.

Problèmes de création de flotte

Lorsqu'une flotte est créée, le GameLift service Amazon lance un flux de travail qui déploie une nouvelle instance dans chacun des emplacements de la flotte et la prépare à faire fonctionner vos serveurs de jeu. Pour une description détaillée, reportez-vous à la sectionComment fonctionne la création GameLift de flottes Amazon. Une flotte ne peut pas héberger de sessions de jeu ni de joueurs tant qu'elle n'a pas atteint le statut Actif. Cette section traite des problèmes les plus courants qui empêchent les flottes de devenir actives.

Téléchargement et validation

Au cours de cette phase, la création du parc peut échouer si des problèmes surviennent avec les fichiers de construction extraits, si le script d'installation ne s'exécute pas ou si le ou les exécutables désignés dans la configuration d'exécution ne sont pas inclus dans les fichiers de construction. Amazon GameLift fournit des journaux relatifs à chacun de ces problèmes.

Si les entrées n'affichent aucun problème, il est possible que le problème soit dû à une erreur de service interne. Si c'est le cas, réessayez de créer la flotte. Si le problème persiste, envisagez de recharger la build de jeu (dans le cas où les fichiers ont été corrompues). Vous pouvez également contacter l'GameLiftassistance Amazon ou poser une question sur le forum.

Création

Les problèmes qui provoquent l'échec lors de la phase de génération sont en général toujours dû aux fichiers de la build du jeu et/ou au script d'installation. Vérifiez que les fichiers de compilation de votre jeu, tels qu'ils ont été chargés sur AmazonGameLift, peuvent être installés sur une machine exécutant le système d'exploitation approprié. Veillez à utiliser une installation de système d'exploitation, et non un environnement de développement existant.

Activation

Les problèmes les plus courants de création de flotte surviennent lors de la phase d'activation. Au cours de cette phase, un certain nombre d'éléments sont en cours de test, notamment la viabilité du serveur de jeu, les paramètres de configuration du runtime et la capacité du serveur de jeu à interagir avec le GameLift service Amazon à l'aide du SDK du serveur. Problèmes courants pouvant survenir lors de l'activation de la flotte :

Les processus serveur ne parviennent pas à démarrer.

> Vérifiez d'abord que vous avez correctement défini le chemin de lancement et les paramètres de lancement facultatifs dans la configuration d'exécution de la flotte. Vous pouvez consulter la configuration d'exécution actuelle du parc en utilisant la section Détails (page détaillée du parc) ou en appelant la AWS CLI commande describe-runtime-configuration. Si la configuration d'exécution semble correcte, vérifiez s'il y a des problèmes avec vos fichiers de la build du jeu et/ou le script d'installation.

Les processus serveur démarrent mais la flotte ne parvient pas à s'activer.

> Si les processus du serveur démarrent et s'exécutent correctement, mais que le parc ne passe pas à l'état Actif, cela est probablement dû au fait que le processus du serveur n'informe pas Amazon GameLift qu'il est prêt à héberger des sessions de jeu. Vérifiez que votre serveur de jeux appelle correctement l'action d'API Server `ProcessReady()` (voir Initialiser le processus du serveur).

La demande de connexion d'appairage de VPC a échoué.

> Pour les flottes créées avec une connexion d'appairage de VPC (voir Pour configurer l'appairage de VPC avec une nouvelle flotte), l'appairage de VPC est effectué au cours de cette phase

d'activation. Si un appairage de VPC échoue pour une raison quelconque, la nouvelle flotte n'obtiendra pas le statut Actif. Vous pouvez suivre le succès ou l'échec de la demande de peering en appelant describe-vpc-peering-connections. Assurez-vous qu'il existe une autorisation d'appairage VPC valide (describe-vpc-peering-authorizations, car les autorisations ne sont valables que pendant 24 heures).

Problèmes liés aux processus serveur

Les processus serveur démarrent mais échouent rapidement ou indiquent une intégrité médiocre.

Hormis les problèmes liés à votre version de génération de jeu, cela peut se produire lorsque vous essayez d'exécuter trop de processus serveur simultanément sur l'instance. Le nombre optimal de processus simultanés dépend des exigences relatives au type d'instance et aux ressources de votre serveur de jeux. Essayez de réduire le nombre de processus simultanés, lequel est défini dans la configuration d'exécution de la flotte, pour voir si les performances s'améliorent. Vous pouvez modifier la configuration d'exécution d'un parc à l'aide de la GameLift console Amazon (modifiez les paramètres d'allocation de capacité du parc) ou en appelant la AWS CLI commande update-runtime-configuration.

Problèmes de suppression de flotte

Une flotte ne peut pas être résiliée en raison d'un nombre d'instances maximum.

Le message d'erreur indique que la flotte en cours de suppression dispose encore d'instances actives, ce qui n'est pas autorisé. Vous devez d'abord mettre à l'échelle une flotte en la réduisant à zéro instance active. Cela s'effectue en définissant manuellement le nombre d'instances souhaitées de la flotte à « 0 », puis en attendant que la réduction prenne effet. Veillez à désactiver la scalabilité automatique, qui contrecarrera les paramètres manuels.

Les actions de VPC ne sont pas autorisées.

Ce problème s'applique uniquement aux flottes pour lesquelles vous avez spécifiquement créé des connexions d'appairage VPC (voir. Peering VPC pour Amazon GameLift Ce scénario se produit car le processus de suppression d'un parc inclut également la suppression du VPC du parc et de toutes les connexions d'appairage VPC. Vous devez d'abord obtenir une autorisation en appelant l'API du GameLift service Amazon CreateVpcPeeringAuthorization() ou en utilisant la commande AWS `create-vpc-peering-authorization` CLI. Une fois que vous avez l'autorisation, vous pouvez supprimer la flotte.

Problèmes liés à la flotte de serveurs en temps réel

Sessions de jeu zombie : elles commencent à exécuter un jeu, mais ne se terminent jamais.

Vous pouvez observer ces problèmes dans les scénarios suivants :

- Les mises à jour des scripts ne sont pas récupérées par les serveurs en temps réel de la flotte.
- La flotte atteint rapidement sa capacité maximale et n'est pas réduite lorsque l'activité des joueurs (telle que les nouvelles demandes de session de jeu) diminue.

Cela est presque certainement dû à l'échec de l'appel de votre script `processEnding` en temps réel. La flotte est activée et les sessions de jeu démarrées, mais il n'y a pas de méthode pour les arrêter. Par conséquent, le serveur en temps réel qui exécute la session de jeu n'est jamais libéré pour en démarrer une nouvelle, et les nouvelles sessions de jeu ne peuvent démarrer que lorsque de nouveaux serveurs en temps réel sont ouverts. De plus, les mises à jour du script en temps réel n'ont aucune incidence sur les sessions de jeu déjà en cours, mais uniquement sur celles qui sont en cours d'exécution.

Pour éviter que cela se produise, les scripts doivent fournir un mécanisme pour déclencher un appel `processEnding`. Comme illustré dans Exemple de script de serveurs en temps réel, une solution consiste à programmer un délai d'expiration de session inactive, tel que si aucun joueur n'est connecté pendant un certain temps, le script met un terme à la session de jeu actuelle.

Toutefois, si vous tombez dans ce scénario, il existe quelques solutions pour débloquer vos serveurs en temps réel. L'astuce consiste à déclencher le redémarrage des processus du serveur en temps réel, ou des instances de flotte sous-jacentes. Dans ce cas, ferme GameLift automatiquement les sessions de jeu pour vous. Une fois que les serveurs en temps réel sont libérés, ils peuvent démarrer de nouvelles sessions de jeu à l'aide de la dernière version du script Realtime.

Il existe deux méthodes pour y parvenir, en fonction de l'ampleur du problème :

- Mettre à l'échelle la totalité de la flotte en la réduisant. Cette méthode est la plus simple à mettre en œuvre mais a un effet étendu. Mettez à l'échelle la flotte en la réduisant à zéro instance, attendez la réduction complète de la flotte, puis remettez-la à l'échelle en l'augmentant. Cela effacera toutes les sessions de jeu existantes et vous permettra de repartir à zéro avec le script Realtime le plus récemment mis à jour.
- Accéder à distance à l'instance et redémarrer le processus. Il s'agit d'une bonne option si vous n'avez que quelques processus à corriger. Si vous êtes déjà connecté à l'instance, par exemple

aux journaux de processus ou de débogage, cette méthode peut être la plus rapide. Consultez Connectez-vous à distance aux instances GameLift de flotte Amazon.

Si vous choisissez de ne pas inclure le mode d'appel `processEnding` dans votre script en temps réel, certaines situations délicates peuvent survenir même lorsque la flotte est active et que des sessions de jeu démarrent. Tout d'abord, une session de jeu en cours d'exécution ne se termine pas. Par conséquent, le processus serveur qui exécute cette session de jeu n'est jamais libre de démarrer une nouvelle session de jeu. Deuxièmement, le serveur Realtime ne récupère aucune mise à jour de script.

Connectez-vous à distance aux instances GameLift de flotte Amazon

Vous pouvez vous connecter à n'importe quelle instance de vos flottes EC2 GameLift gérées par Amazon actives. Les raisons les plus courantes pour accéder à une instance sont les suivantes :

- Résoudre les problèmes liés à l'intégration de votre serveur de jeu
- Ajustez la configuration de votre environnement d'exécution et les autres paramètres spécifiques au parc
- Obtenez l'activité du serveur de jeu en temps réel, par exemple le suivi des journaux.
- Exécutez des outils d'analyse comparative en utilisant le trafic réel des joueurs.
- Étudiez les problèmes spécifiques liés à une session de jeu ou à un processus serveur.

Lorsque vous vous connectez à une instance, prenez en compte les problèmes potentiels suivants :

- Vous pouvez vous connecter aux instances des flottes actives. Les flottes non actives, celles qui s'activent ou sont en état d'erreur, peuvent être accessibles pendant une courte période. Pour obtenir de l'aide concernant les problèmes d'activation de la flotte, consultezRésoudre les problèmes liés à la GameLift flotte Amazon.
- La connexion à une instance active n'affecte pas l'activité d'hébergement de l'instance. L'instance continue de démarrer et d'arrêter les processus du serveur en fonction de la configuration d'exécution. Il active et héberge une session de jeu. Il peut s'arrêter en réponse à un événement de réduction ou à un autre événement.
- Toute modification apportée aux fichiers ou aux paramètres de l'instance peut avoir un impact sur les sessions de jeu actives de l'instance et sur les joueurs connectés.

Les instructions suivantes décrivent comment se connecter à distance à une instance à l'aide de l'interface de ligne de AWS commande (CLI). Vous pouvez également effectuer des appels programmatiques à l'aide du AWS SDK, comme indiqué dans la référence de l'API de GameLift service Amazon.

Collectez les données d'instance

Collectez les informations suivantes :

- L'ID de l'instance à laquelle vous souhaitez vous connecter. Vous pouvez utiliser l'ID d'instance ou l'ARN.
- Version GameLift du SDK du serveur Amazon utilisée sur l'instance. Le SDK du serveur est intégré à la version du jeu exécutée sur l'instance.

Pour récupérer les données de l'instance

Les étapes suivantes supposent que vous disposez d'un ID de flotte EC2 géré pour l'instance à laquelle vous souhaitez vous connecter.

1. Obtenez le nom du calcul.

   Appelez list-compute pour le parc EC2 géré afin d'obtenir la liste de tous les ordinateurs actifs du parc. Pour une flotte à site unique, spécifiez l'ID ou l'ARN de la flotte. Pour une flotte multisite, spécifiez l'ID ou l'ARN de la flotte et un emplacement. Pour un parc EC2 géré, les calculs sont des instances EC2 et la propriété renvoyée `ComputeName` est l'ID de l'instance. Par exemple :

   Demande

   ```
   aws gamelift list-compute \
     --fleet-id  "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa" \
     --location ""sa-east-1"
   ```

   Réponse

   ```
   {
     "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. Trouvez la version du SDK du serveur.

   La version du SDK du serveur est un attribut d'une ressource de build.

   a. Appelez describe-fleet-attributesavec un identifiant de flotte pour obtenir l'identifiant de build et l'ARN de la flotte.

   b. Appelez describe-build avec l'ID de build ou l'ARN pour obtenir la version du SDK du serveur de la version.

      Par exemple :

      Demande

```
aws gamelift describe-fleet-attributes /
  --fleet-ids  "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa"
```

   Réponse

```
{
  "FleetAttributes": [
    {
      "FleetId": "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
      "ComputeType": "EC2",
      "BuildId": "build-3333cccc-44dd-55ee-66ff-00001111aa22",
      . . .
    }
  ]
```

```
}
```

Demande

```
aws gamelift describe-build /
  --build-id "build-3333cccc-44dd-55ee-66ff-00001111aa22"
```

Réponse

```
"Build": {
  "BuildId": "build-1111aaaa-22bb-33cc-44dd-5555eeee66ff",
  "Name": "My_Game_Server_Build_One",
  "OperatingSystem": "AMAZON_LINUX_2",
  "ServerSdkVersion": "5.1.1",
  . . .
}
```

Se connecter à une instance (SDK 5 du serveur)

Si l'instance à laquelle vous souhaitez vous connecter exécute une version de jeu avec la version 5.x du SDK du serveur, suivez les instructions suivantes pour vous connecter à l'instance à l'aide d'Amazon EC2 Systems Manager (SSM). Vous pouvez accéder à des instances à distance qui exécutent Windows ou Linux.

1. Demandez des informations d'identification d'accès pour l'instance. Lorsque vous avez un nom de calcul et un ID de flotte pour l'instance à laquelle vous souhaitez vous connecter, appelez get-compute-access. En cas de succès, Amazon GameLift renvoie un ensemble d'informations d'identification temporaires pour accéder à l'instance. Par exemple :

   Demande

   ```
   aws gamelift get-compute-access \
   --compute-name i-11111111a222b333c \
   --fleet-id fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa
   --region us-west-2
   ```

   Réponse

   ```
   {
   ```

```
  "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. Exportez les informations d'accès. Vous pouvez éventuellement exporter les informations d'identification vers des variables d'environnement et les utiliser pour configurer la AWS CLI pour l'utilisateur par défaut. Pour plus de détails, consultez la section Variables d'environnement pour configurer la AWS CLI dans le guide de AWS Command Line Interface l'utilisateur.

```
export AWS_ACCESS_KEY_ID=ASIAIOSFODNN7EXAMPLE
export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
export AWS_SESSION_TOKEN=AQoDYXdzEJr...<remainder of session token>
```

3. Connectez-vous à l'instance de flotte. Démarrez une session SSM avec l'instance à laquelle vous souhaitez vous connecter. Incluez la AWS région ou l'emplacement de l'instance. Pour plus d'informations, consultez la section Démarrage d'une session (AWSCLI) dans le guide de l'utilisateur d'Amazon EC2 Systems Manager. Utilisez les informations d'identification que vous avez acquises à l'étape 1. Par exemple :

```
aws ssm start-session \
--target i-11111111a222b333c \
--region us-west-2
```

Connect à une instance (SDK de serveur 4.x ou version antérieure)

Si l'instance à laquelle vous souhaitez vous connecter exécute une version de jeu avec la version 4 ou une version antérieure du SDK du serveur, suivez les instructions ci-dessous. Vous pouvez vous connecter à des instances qui exécutent Windows ou Linux. Connectez-vous à une instance Windows à l'aide d'un client RDP (Remote Desktop Protocol). Connectez-vous à une instance Linux à l'aide d'un client SSH.

1. Demandez des informations d'identification d'accès pour l'instance. Lorsque vous disposez d'un ID d'instance, utilisez la commande get-instance-accesspour demander des informations

d'accès. En cas de succès, Amazon GameLift renvoie le système d'exploitation, l'adresse IP et un ensemble d'informations d'identification (nom d'utilisateur et clé secrète) de l'instance. Le format des informations d'identification dépend du système d'exploitation de l'instance. Utilisez les instructions suivantes extraire les informations d'identification pour RDP ou SSH.

- Pour les instances Windows : pour se connecter à une instance Windows, RDP a besoin d'un nom d'utilisateur et d'un mot de passe. La demande `get-instance-access` renvoie ces valeurs sous la forme de chaînes simples, afin que vous puissiez les valeurs telles quelles. Exemple d'informations d'identification :

```
"Credentials": {
    "Secret": "aA1bBB2cCCd3EEE",
    "UserName": "gl-user-remote"
}
```

- Pour les instances Linux : pour se connecter à une instance Linux, SSH a besoin d'un nom d'utilisateur et d'une clé privée. Amazon GameLift émet des clés privées RSA et les renvoie sous forme de chaîne unique, le caractère de nouvelle ligne (\n) indiquant les sauts de ligne. Pour rendre la clé privée utilisable, procédez comme suit : (1) convertissez la chaîne en `.pem` fichier et (2) définissez les autorisations pour le nouveau fichier. Exemple d'informations d'identification renvoyées :

```
"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"
}
```

Lorsque vous utilisez la AWS CLI, vous pouvez générer automatiquement un `.pem` fichier en incluant les paramètres --query et --output dans votre `get-instance-access` requête.

Pour définir des autorisations sur le fichier `.pem`, exécutez la commande suivante :

```
$ chmod 400 MyPrivateKey.pem
```

2. Ouvrez un port pour la connexion à distance. Vous pouvez accéder aux instances des GameLift flottes Amazon via n'importe quel port autorisé dans la configuration de la flotte. Vous pouvez afficher les paramètres de port de la flotte à l'aide de la commande [describe-fleet-port-settings](#).

   La bonne pratique consiste à ouvrir des ports pour l'accès à distance uniquement lorsque vous en avez besoin et à les fermer lorsque vous avez terminé. Vous ne pouvez pas mettre à jour les paramètres du port après avoir créé une flotte, mais avant qu'elle ne soit active. Si vous êtes bloqué, recréez la flotte avec les paramètres du port ouverts.

   Utilisez la commande [update-fleet-port-settings](#) pour ajouter un paramètre de port pour la connexion à distance (22 pour SSH ou 3389 pour RDP, par exemple). Pour la valeur de plage d'adresses IP, spécifiez les adresses IP pour les périphériques que vous voulez utiliser pour la connexion (converties au format CIDR). Exemple :

```
$ AWS gamelift update-fleet-port-settings
    --fleet-id  "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa"
    --inbound-permission-authorizations
 "FromPort=22,ToPort=22,IpRange=54.186.139.221/32,Protocol=TCP"
```

   L'exemple suivant ouvre le port 3389 sur un parc 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. Ouvrez un client de connexion à distance. Utilisez le Bureau à distance pour Windows ou SSH pour les instances Linux. Connectez-vous à l'instance à l'aide de l'adresse IP, du paramètre de port et des informations d'identification d'accès.

   Exemple SSH :

   ```
   ssh -i MyPrivateKey.pem gl-user-remote@192.0.2.0
   ```

Afficher des fichiers sur des instances distantes

Lorsque vous êtes connecté à une instance à distance, vous disposez d'un accès utilisateur et administratif complet. Cela signifie que vous avez également la possibilité de produire des erreurs et des défaillances dans l'hébergement de jeux. Si l'instance héberge des parties avec des joueurs actifs, vous courez le risque de bloquer des sessions de jeu et de perdre des joueurs, ou de perturber les processus d'arrêt du jeu et de provoquer des erreurs dans les données et les journaux de jeu enregistrés.

Recherchez les ressources suivantes sur une instance d'hébergement :

- Fichiers de build de jeu. Ces fichiers sont le build du jeu que vous avez chargé sur Amazon GameLift. Ils incluent un ou plusieurs exécutables, actifs et dépendances du serveur de jeu. Les fichiers de compilation du jeu se trouvent dans un répertoire racine appelé `game` :
  - Sous Windows : `c:\game`
  - Sous Linux : `/local/game`
- Fichiers journaux de jeu. Trouvez les fichiers journaux générés par votre serveur de jeu dans le répertoire `game` racine, quel que soit le chemin de répertoire que vous avez indiqué.
- Ressources GameLift d'hébergement Amazon. Le répertoire racine `Whitewater` contient les fichiers utilisés par le GameLift service Amazon pour gérer l'activité d'hébergement de jeux. Ne modifiez pas ces fichiers pour quelque raison que ce soit.
- Configuration d'exécution. N'accédez pas à la configuration d'exécution pour des instances individuelles. Pour modifier une propriété de configuration d'exécution, mettez à jour la configuration d'exécution du parc (voir le fonctionnement du AWS SDK UpdateRuntimeConfigurationou le AWS CLI update-runtime-configuration).

- Données relatives à la flotte. Un fichier JSON contient des informations sur le parc auquel appartient l'instance, destinées aux processus du serveur exécutés sur l'instance. Le fichier JSON se trouve à l'emplacement suivant :
  - Sous Windows : `C:\GameMetadata\gamelift-metadata.json`
  - Sous Linux : `/local/gamemetadata/gamelift-metadata.json`
- Certificat TLS Si l'instance fait partie d'un parc sur lequel la génération de certificats TLS est activée, recherchez les fichiers de certificat, y compris le certificat, la chaîne de certificats, la clé privée et le certificat racine à l'emplacement suivant :
  - Sous Windows : `c:\\GameMetadata\Certificates`
  - Sous Linux : `/local/gamemetadata/certificates/`

Élargir la capacité GameLift d'hébergement d'Amazon

La capacité d'hébergement, mesurée en instances, représente le nombre de sessions de jeu qu'Amazon GameLift peut héberger simultanément et le nombre de joueurs simultanés que ces sessions de jeu peuvent accueillir. L'une des tâches les plus difficiles de l'hébergement de jeux consiste à adapter la capacité pour répondre à la demande des joueurs sans gaspiller d'argent sur des ressources dont vous n'avez pas besoin. Pour plus d'informations, veuillez consulter Dimensionnement de la capacité d'une flotte.

La capacité est ajustée au niveau de l'emplacement de la flotte. Toutes les flottes ont au moins un emplacement : la AWS région d'origine de la flotte. Lors de la visualisation ou de la mise à l'échelle de la capacité, les informations sont répertoriées par emplacement, y compris la région d'origine de la flotte et tout autre site distant.

Vous pouvez définir manuellement le nombre d'instances à gérer ou configurer une mise à l'échelle automatique pour ajuster dynamiquement la capacité en fonction de l'évolution de la demande des joueurs. Nous vous recommandons de commencer par activer l'option de mise à l'échelle automatique basée sur la cible. L'objectif de la mise à l'échelle automatique basée sur les objectifs est de maintenir suffisamment de ressources d'hébergement pour répondre aux besoins des joueurs actuels, plus un peu plus pour faire face à des pics imprévus de la demande des joueurs. Pour la plupart des jeux, la mise à l'échelle automatique basée sur les cibles constitue une solution de mise à l'échelle très efficace.

Les rubriques de cette section fournissent une aide détaillée sur les tâches suivantes :

- Définir des limites de capacité maximale et minimale pour le dimensionnement des capacités

- Définir manuellement des niveaux de capacité
- Utiliser la mise à l'échelle automatique basée sur les cibles
- Gestion de la mise à l'échelle automatique basée sur des règles (fonctionnalité avancée)
- Désactiver temporairement la mise à l'échelle automatique

Vous pouvez effectuer la plupart des activités de dimensionnement de votre flotte à l'aide de la GameLift console Amazon. Vous pouvez également utiliser un AWS SDK ou le AWS Command Line Interface (AWS CLI) avec l'API du GameLift service Amazon.

Pour gérer la capacité du parc dans la console

1. Ouvrez la GameLiftconsole Amazon.
2. Dans le volet de navigation, choisissez Hosting, Fleets.
3. Sur la page Flottes, choisissez le nom d'une flotte active pour ouvrir la page détaillée de la flotte.
4. Choisissez l'onglet Scaling. Dans cet onglet, vous pouvez :

    - Consultez les mesures de mise à l'échelle historiques pour l'ensemble du parc.
    - Affichez et mettez à jour les paramètres de capacité pour chaque emplacement de flotte, y compris les limites de dimensionnement et les paramètres de capacité actuels.
    - Mettez à jour le dimensionnement automatique basé sur les cibles, consultez les politiques de dimensionnement automatique basées sur des règles appliquées à l'ensemble du parc et suspendez l'activité de dimensionnement automatique pour chaque site.

Rubriques

- Définissez les limites GameLift de capacité d'Amazon
- Configuration manuelle de la capacité d'une GameLift flotte Amazon
- Adaptez automatiquement la capacité de votre flotte avec Amazon GameLift

Définissez les limites GameLift de capacité d'Amazon

Lorsque vous augmentez la capacité d'hébergement d'un emplacement de GameLift flotte Amazon, manuellement ou automatiquement, tenez compte des limites de dimensionnement de l'emplacement. Tous les emplacements de la flotte sont soumis à des limites minimale et maximale qui définissent la plage autorisée pour la capacité de l'emplacement. Par défaut, les limites relatives

aux emplacements de flotte sont limitées à 0 instance au minimum et à 1 instance au maximum. Avant de pouvoir redimensionner l'emplacement d'une flotte, ajustez les limites.

Si vous utilisez la mise à l'échelle automatique, la limite maximale permet GameLift à Amazon d'étendre l'emplacement d'une flotte pour répondre à la demande des joueurs, tout en évitant des coûts d'hébergement exorbitants, par exemple lors d'une attaque DDOS. Configurez une CloudWatchalarme Amazon pour vous avertir lorsque la capacité approche de la limite maximale, afin que vous puissiez évaluer la situation et l'ajuster manuellement si nécessaire. (Vous pouvez également créer une alarme de facturation pour surveiller AWS les coûts.) La limite minimale est utile pour maintenir la disponibilité de l'hébergement, même lorsque la demande des joueurs est faible.

Vous pouvez définir des limites de capacité pour les emplacements d'une flotte dans la GameLiftconsole Amazon ou en utilisant le AWS Command Line Interface (AWS CLI).

Pour définir les limites de capacité

Console

1. Ouvrez la GameLiftconsole Amazon.
2. Dans le volet de navigation, choisissez Hosting, Fleets.
3. Sur la page Flottes, choisissez le nom d'une flotte active pour ouvrir la page détaillée de la flotte.
4. Dans l'onglet Mise à l'échelle, sous Dimensionnement de la capacité, sélectionnez un emplacement de flotte, puis choisissez Modifier.
5. Dans la boîte de dialogue Modifier la capacité de dimensionnement, définissez le nombre d'instances pour la taille minimale, les instances souhaitées et la taille maximale.
6. Choisissez Confirm (Confirmer).

AWS CLI

1. Vérifiez les paramètres de capacité actuels. Dans une fenêtre de ligne de commande, utilisez la describe-fleet-location-capacitycommande avec l'ID du parc et l'emplacement pour lesquels vous souhaitez modifier la capacité. Cette commande renvoie un FleetCapacityobjet qui inclut les paramètres de capacité actuels de l'emplacement. Déterminez si les nouvelles limites d'instances peuvent s'adapter au paramètre d'instances actuellement souhaité.

```
aws gamelift describe-fleet-location-capacity \
    --fleet-id <fleet identifier> \
```

```
    --location <location name>
```

2. Mettez à jour les paramètres de limite. Dans une fenêtre de ligne de commande, utilisez la update-fleet-capacitycommande avec les paramètres suivants. Vous pouvez ajuster les deux limites d'instance et le nombre d'instances souhaitées avec la même commande.

```
--fleet-id <fleet identifier>
--location <location name>
--max-size <maximum capacity for scaling>
--min-size <minimum capacity for scaling>
--desired-instances <fleet capacity goal>
```

Exemple :

```
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
```

Si votre demande aboutit, Amazon GameLift renvoie l'identifiant de la flotte. Si la nouvelle `min-size` valeur `max-size` ou la valeur est en conflit avec le `desired-instances` paramètre actuel, Amazon GameLift renvoie une erreur.

Configuration manuelle de la capacité d'une GameLift flotte Amazon

Lorsque vous créez une nouvelle flotte, Amazon définit GameLift automatiquement les instances souhaitées sur une instance dans chaque emplacement de flotte. Amazon GameLift déploie ensuite une nouvelle instance sur chaque site. Pour modifier la capacité du parc, vous pouvez ajouter une politique de dimensionnement automatique basée sur des cibles, ou vous pouvez définir manuellement le nombre d'instances que vous souhaitez pour un site. Pour plus d'informations, veuillez consulter Dimensionnement de la capacité d'une flotte.

La définition manuelle de la capacité d'un parc peut s'avérer utile lorsque vous n'avez pas besoin d'une mise à l'échelle automatique ou lorsque vous devez maintenir la capacité à un niveau spécifié. La définition manuelle de la capacité ne fonctionne que si vous n'utilisez pas de politique de mise à l'échelle automatique basée sur des cibles. Si vous disposez d'une politique de dimensionnement

automatique basée sur des cibles, elle réinitialise immédiatement la capacité souhaitée en fonction de ses propres règles de dimensionnement.

Vous pouvez définir manuellement la capacité dans la GameLift console Amazon ou à l'aide du AWS Command Line Interface (AWS CLI). Le statut de la flotte doit être actif.

Suspendre le dimensionnement automatique

Vous pouvez suspendre toutes les activités de mise à l'échelle automatique pour chaque emplacement du parc. Lorsque le dimensionnement automatique est suspendu, le nombre d'instances souhaité dans l'emplacement du parc reste le même, à moins qu'il ne soit modifié manuellement. Lorsque vous suspendez le dimensionnement automatique pour un site, cela a une incidence sur les politiques actuelles de la flotte et sur toutes les politiques que vous pourriez définir à l'avenir.

Pour définir manuellement la capacité de la flotte

Console

1. Ouvrez la GameLiftconsole Amazon.
2. Dans le volet de navigation, choisissez Hosting, Fleets.
3. Sur la page Flottes, choisissez le nom d'une flotte active pour ouvrir la page détaillée de la flotte.
4. Dans l'onglet Mise à l'échelle, sous Emplacements de mise à l'échelle automatique suspendus, sélectionnez chaque emplacement pour lequel vous souhaitez suspendre le dimensionnement automatique, puis choisissez Suspendre.
5. Sous Dimensionnement de la capacité, sélectionnez un emplacement que vous souhaitez définir manuellement, puis choisissez Modifier.
6. Dans la boîte de dialogue Modifier la capacité de dimensionnement, définissez votre valeur préférée pour les instances souhaitées, puis choisissez Confirmer. Cela indique à Amazon GameLift le nombre d'instances à maintenir actives, prêtes à héberger des sessions de jeu.

   Amazon GameLift répond aux modifications en déployant des instances supplémentaires ou en fermant celles qui ne sont pas nécessaires. Au fur et à mesure qu'Amazon GameLift termine ce processus, le nombre d'instances actives dans l'emplacement change pour correspondre à la valeur d'instances souhaitée mise à jour. Ce processus peut prendre un certain temps.

AWS CLI

1. Vérifiez les paramètres de capacité actuels. Dans une fenêtre de ligne de commande, utilisez la describe-fleet-location-capacitycommande avec l'ID du parc et l'emplacement pour lesquels vous souhaitez modifier la capacité. Cette commande renvoie un FleetCapacityobjet qui inclut les paramètres de capacité actuels de l'emplacement. Déterminez si les limites d'instances peuvent s'adapter au nouveau paramètre d'instances souhaité.

   ```
   aws gamelift describe-fleet-location-capacity \
       --fleet-id <fleet identifier> \
       --location <location name>
   ```

2. Mettez à jour la capacité souhaitée. Utilisez la update-fleet-capacitycommande avec l'ID du parc, l'emplacement et une nouvelle valeur pour les instances souhaitées. Si cette valeur se situe en dehors de la plage limite actuelle, vous pouvez ajuster les valeurs limites dans la même commande.

   ```
   --fleet-id <fleet identifier>
   --location <location name>
   --desired-instances <fleet capacity as an integer>
   --max-size <maximum capacity>    [Optional]
   --min-size <minimum capacity>    [Optional]
   ```

   Exemple :

   ```
   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
   ```

Si votre demande aboutit, Amazon GameLift renvoie l'identifiant de la flotte. Si le nouveau paramètre d'instances souhaité dépasse les limites minimale et maximale, Amazon GameLift renvoie un message d'erreur.

Adaptez automatiquement la capacité de votre flotte avec Amazon GameLift

Utilisez la mise à l'échelle automatique sur Amazon GameLift pour faire évoluer de manière dynamique la capacité de votre flotte en fonction de l'activité des serveurs de jeux. Au fur et à mesure que les joueurs arrivent et commencent des sessions de jeu, le dimensionnement automatique peut ajouter d'autres instances ; à mesure que la demande des joueurs diminue, le dimensionnement automatique peut mettre fin à des instances inutiles. La mise à l'échelle automatique est un moyen efficace de minimiser vos ressources et vos coûts d'hébergement, tout en offrant une expérience de jeu fluide et rapide.

Pour utiliser la mise à l'échelle automatique, vous créez des politiques de mise à l'échelle qui indiquent à Amazon GameLift quand augmenter ou diminuer. Il existe deux types de politiques de dimensionnement : basées sur des cibles et basées sur des règles. L'approche basée sur les cibles, à savoir le suivi des cibles, est une solution complète. Nous le recommandons comme l'option la plus simple et la plus efficace. Les politiques de dimensionnement basées sur des règles exigent que vous définissiez chaque aspect du processus décisionnel relatif au dimensionnement automatique, ce qui est utile pour résoudre des problèmes spécifiques. Cette solution fonctionne mieux en complément de la mise à l'échelle automatique basée sur les cibles.

Vous pouvez gérer le dimensionnement automatique basé sur les cibles à l'aide de la GameLift console Amazon, du AWS Command Line Interface (AWS CLI) ou d'un AWS SDK. Vous pouvez gérer le dimensionnement automatique basé sur des règles uniquement à l'aide du SDK AWS CLI ou d'un AWS SDK, mais vous pouvez consulter les politiques de dimensionnement basées sur des règles dans la console.

Rubriques

Mise à l'échelle automatique basée sur les cibles

La mise à l'échelle automatique basée sur les objectifs pour Amazon GameLift ajuste les niveaux de capacité en fonction des indicateurs du parc. `PercentAvailableGameSessions` Cette métrique représente la réserve disponible de la flotte en cas d'augmentation soudaine de la demande des joueurs.

La raison principale du maintien d'une capacité de mémoire tampon est le temps d'attente du joueur. Lorsque les machines à sous des sessions de jeu sont prêtes et qu'elles attendent, il faut

quelques secondes pour que de nouveaux joueurs puissent participer aux sessions de jeu. Si aucune ressource n'est disponible, les joueurs doivent patienter que les sessions de jeu existantes se terminent ou que de nouvelles ressources deviennent disponibles. Le démarrage de nouvelles instances et de nouveaux processus de serveur peut prendre quelques minutes.

Lorsque vous configurez une mise à l'échelle automatique basée sur des cibles, spécifiez la taille de la zone tampon que vous souhaitez que le parc conserve. Comme elle `PercentAvailableGameSessions` mesure le pourcentage des ressources disponibles, la taille réelle de la zone tampon est un pourcentage de la capacité totale de la flotte. Amazon GameLift ajoute ou supprime des instances pour conserver la taille de la mémoire tampon cible. Avec une mémoire tampon importante, vous réduisez le temps d'attente, mais vous payez également pour des ressources supplémentaires que vous n'utiliserez peut-être pas. Si vos joueurs sont plus tolérant en ce qui concerne les temps d'attente, vous pouvez réduire les coûts en définissant un tampon plus petit.

Pour définir une mise à l'échelle automatique basée sur les cibles

Console

1. Ouvrez la GameLiftconsole Amazon.
2. Dans le volet de navigation, choisissez Hosting, Fleets.
3. Sur la page Flottes, choisissez le nom d'une flotte active pour ouvrir la page détaillée de la flotte.
4. Choisissez l'onglet Scaling. Cet onglet affiche les métriques de mise à l'échelle de l'historique du parc et contient des contrôles pour l'ajustement des paramètres de mise à l'échelle actuels.
5. Sous Dimensionnement de la capacité, vérifiez que les limites de taille minimale et de taille maximale sont appropriées pour le parc. Lorsque la mise à l'échelle automatique est activée, la capacité s'ajuste entre ces deux limites.
6. Dans la politique de mise à l'échelle automatique basée sur Target, choisissez Modifier.
7. Dans la boîte de dialogue Modifier la politique de mise à l'échelle automatique basée sur les cibles, pour Pourcentage de sessions de jeu disponibles, définissez le pourcentage que vous souhaitez conserver, puis choisissez Confirmer. Une fois que vous avez confirmé les paramètres, Amazon GameLift ajoute une nouvelle politique basée sur les cibles sous Politique de mise à l'échelle automatique basée sur les cibles.

AWS CLI

1. Définissez les limites de capacité. Définissez les valeurs limites à l'aide de la update-fleet-capacitycommande. Pour plus d'informations, veuillez consulter Définissez les limites GameLift de capacité d'Amazon.
2. Créez une stratégie. Ouvrez une fenêtre de ligne de commande et utilisez la put-scaling-policycommande avec les paramètres de votre politique. Pour mettre à jour une stratégie existante, spécifiez le nom de la stratégie et indiquez une version complète de la stratégie mise à jour.

```
--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>
```

Exemple :

```
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"
```

Mise à l'échelle automatique avec des politiques basées sur des règles

Les politiques de dimensionnement basées sur des règles d'Amazon GameLift offrent un contrôle précis lors du dimensionnement automatique de la capacité d'une flotte en réponse à l'activité des joueurs. Pour chaque politique, vous pouvez associer la mise à l'échelle à l'une des nombreuses mesures de la flotte, identifier un point de déclenchement et personnaliser l'événement d'augmentation ou de réduction d'échelle correspondant. Les politiques basées sur des règles sont utiles pour compléter la mise à l'échelle basée sur des objectifs afin de faire face à des circonstances particulières.

Une politique basée sur des règles stipule ce qui suit : « Si un indicateur de flotte atteint ou dépasse une valeur seuil pendant un certain temps, modifiez la capacité de la flotte d'une quantité spécifiée. »

Cette rubrique décrit la syntaxe utilisée pour construire une déclaration de stratégie et fournit une aide pour la création et la gestion de vos règles basées sur des règles.

Gérer les stratégies basées sur des règles

Créez, mettez à jour ou supprimez des politiques basées sur des règles à l'aide d'un AWS SDK ou du AWS Command Line Interface (AWS CLI) avec l'API de service Amazon GameLift. Vous pouvez consulter toutes les politiques actives dans la GameLift console Amazon.

Pour arrêter temporairement toutes les politiques de dimensionnement d'une flotte, utilisez la AWS CLI commande stop-fleet-actions.

Pour créer ou mettre à jour une politique de dimensionnement basée sur des règles () AWS CLI :

1. Définissez les limites de capacité. Définissez l'une des valeurs limites ou les deux à l'aide de la update-fleet-capacitycommande. Pour plus d'informations, veuillez consulter Définissez les limites GameLift de capacité d'Amazon.
2. Créez une stratégie. Ouvrez une fenêtre de ligne de commande et utilisez la put-scaling-policycommande avec les paramètres de votre politique. Pour mettre à jour une stratégie existante, spécifiez le nom de la stratégie et indiquez une version complète de la stratégie mise à jour.

   ```
   --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>
   ```

   Exemple :

   ```
   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
```

Pour supprimer une politique de dimensionnement basée sur des règles à l'aide de : AWS CLI

- Ouvrez une fenêtre de ligne de commande et utilisez la delete-scaling-policycommande avec l'ID du parc et le nom de la politique.

  Exemple :

```
aws gamelift delete-scaling-policy \
    --fleet-id fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa \
    --name "Scale up when AGS<50"
```

Syntaxe pour les règles de mise à l'échelle automatique

Pour élaborer une déclaration de politique de dimensionnement basée sur des règles, spécifiez six variables :

Si *<nom de métrique>* demeure *<opérateur de comparaison>* *<valeur de seuil>* pour *<période d'évaluation>*, remplacez la capacité de la flotte en utilisant *<type d'ajustement>* sur/par *<valeur d'ajustement>*.

Par exemple, cette déclaration de politique lance un événement de mise à l'échelle chaque fois que la capacité supplémentaire d'une flotte est inférieure à ce qui est nécessaire pour gérer 50 nouvelles sessions de jeu :

Si `AvailableGameSessions` reste à `less than 50` pendant `10 minutes`, modifier à la capacité de la flotte avec `ChangeInCapacity` par `1 instances`.

Nom de la métrique

Pour démarrer un événement de dimensionnement, associez une politique de dimensionnement automatique à l'une des mesures spécifiques au parc suivantes. Pour une description complète des mesures, reportez-vous à la sectionGameLiftMétriques Amazon pour les flottes.

- Activation de sessions de jeu
- Sessions de jeu actives
- Sessions de jeu disponibles

- Pourcentage de sessions de jeu disponibles
- Instances actives
- Sessions de joueur disponibles
- Sessions de joueur actuelles
- Instances inactives
- Pourcentage d'instances inactives

Si la flotte se trouve dans une file d'attente de session de jeu, vous pouvez utiliser les statistiques suivantes :

- Profondeur de la file d'attente : nombre de demandes de sessions de jeu en attente pour lesquelles cette flotte est la meilleure solution d'hébergement disponible.
- Temps d'attente — Temps d'attente propre à la flotte. Durée pendant laquelle la demande de session de jeu plus ancienne attend d'être traitée. Le temps d'attente d'une flotte correspond à la durée passée en file d'attente de la plus ancienne demande actuelle.

Opérateur de comparaison

Indique à Amazon GameLift comment comparer les données métriques à la valeur du seuil. Les opérateurs de comparaison valides incluent supérieur à (**>**), inférieur à (**= <**), greater than or equal (**>**) et inférieur ou égal (**<=**).

Valeur de seuil

Lorsque la valeur de métrique spécifiée atteint ou dépasse la valeur seuil, un événement de mise à l'échelle démarre. Cette valeur est toujours un nombre entier positif.

Période d'évaluation

La métrique doit atteindre ou dépasser la valeur seuil pendant toute la durée de la période d'évaluation avant de démarrer un événement de dimensionnement. La longueur de la période d'évaluation est consécutive ; si la métrique s'écarte du seuil, la période d'évaluation recommence.

Type et valeur d'ajustement

Cet ensemble de variables fonctionne ensemble pour spécifier la manière dont Amazon GameLift doit ajuster la capacité de la flotte lorsqu'un événement de dimensionnement commence. Choisissez parmi trois types de réglage possibles :

- Modification de la capacité : augmentez ou diminuez la capacité actuelle d'un nombre spécifié d'instances. Définissez la valeur d'ajustement en fonction du nombre d'instances à ajouter ou

à retirer du parc. Les valeurs positives ajoutent des instances, tandis que les valeurs négatives retirent des instances. Par exemple, une valeur de « -10 » permet de réduire le parc de 10 instances, quelle que soit la taille totale du parc.

- Pourcentage de variation de la capacité — Augmentez ou diminuez la capacité actuelle d'un pourcentage spécifié. Définissez la valeur d'ajustement en fonction du pourcentage dont vous souhaitez augmenter ou diminuer la capacité du parc. Les valeurs positives ajoutent des instances, tandis que les valeurs négatives retirent des instances. Par exemple, pour un parc de 50 instances, une variation en pourcentage de « 20 » ajoute 10 instances au parc.
- Capacité exacte : augmentez ou diminuez la capacité actuelle jusqu'à une valeur spécifique. Définissez la valeur d'ajustement sur le nombre exact d'instances que vous voulez maintenir dans le parc.

Conseils pour une mise à l'échelle automatique basée sur des règles

Les suggestions suivantes peuvent vous aider à tirer le meilleur parti du dimensionnement automatique grâce à des politiques basées sur des règles.

Utilisation de plusieurs stratégies

Vous pouvez appliquer simultanément plusieurs politiques de mise à l'échelle automatique pour un parc. Le scénario le plus courant consiste à avoir une stratégie basée sur la cible pour gérer la plupart des besoins de dimensionnement et d'utiliser des stratégies basées sur des règles pour gérer les cas limites. Il n'y a aucune limite à l'utilisation de plusieurs politiques.

Dans le cas de stratégies multiples, chaque stratégie se comporte de manière indépendante. Il n'existe aucun moyen de contrôler la séquence des événements de dimensionnement. Par exemple, si plusieurs règles régissent la mise à l'échelle, il est possible que l'activité des joueurs déclenche simultanément plusieurs événements de mise à l'échelle. Évitez les politiques qui se déclenchent mutuellement. Par exemple, vous pouvez créer une boucle infinie si vous créez des politiques d'augmentation et de réduction qui fixent la capacité au-delà du seuil correspondant l'une à l'autre.

Définir une capacité maximale et minimale

Chaque parc comporte une limite de capacité maximale et minimale. Cette fonctionnalité est importante lors de l'utilisation de la mise à l'échelle automatique. La mise à l'échelle automatique ne définit jamais la capacité à une valeur en dehors de cette plage. Par défaut, les flottes nouvellement créées ont un minimum de 0 et un maximum de 1. Pour que votre politique de dimensionnement automatique affecte la capacité comme prévu, augmentez la valeur maximale.

La capacité du parc est également limitée par les limites du type d'instance du parc et par les quotas de service de votreCompte AWS. Vous ne pouvez pas définir de minimum et de maximum en dehors de ces limites et de ces quotas de compte.

Métriques de suivi après une modification de la capacité

Après avoir modifié la capacité en réponse à une politique de mise à l'échelle automatique, Amazon GameLift attend 10 minutes avant de répondre aux déclencheurs de la même politique. Cette attente donne à Amazon le GameLift temps d'ajouter les nouvelles instances, de lancer les serveurs de jeu, de connecter les joueurs et de commencer à collecter des données à partir des nouvelles instances. Pendant ce temps, Amazon GameLift évalue la politique par rapport à la métrique et suit la période d'évaluation de la politique, qui recommence après un événement de mise à l'échelle. Cela signifie qu'une politique de dimensionnement peut lancer un autre événement de dimensionnement immédiatement après la fin du délai d'attente.

Il n'y a pas de temps d'attente entre les événements de dimensionnement et le début des différentes politiques de dimensionnement automatique.

Configuration des GameLift files d'attente Amazon pour le placement des sessions de jeu

La file d'attente des sessions de jeu est le principal mécanisme permettant de traiter les nouvelles demandes de session de jeu et de localiser les serveurs de jeu disponibles pour les héberger. Les files d'attente offrent des avantages considérables aux développeurs de jeux et aux joueurs. Il s'agit des licences suivantes :

- Les files d'attente offrent le meilleur placement possible. Lors du traitement des demandes de placement de sessions de jeu, une file d'attente utilise les GameLift algorithmes d'Amazon pour hiérarchiser les emplacements des files d'attente en fonction d'un ensemble de préférences définies.
- Organisez des jeux sur des flottes Spot moins chères. Utilisez les files d'attente pour optimiser l'utilisation des flottes AWS Spot, qui offrent des coûts d'hébergement nettement inférieurs. Par défaut, les files d'attente essaient toujours de placer de nouvelles sessions de jeu dans les flottes Spot.
- Créez de nouveaux jeux plus rapidement en cas de forte demande. Les files d'attente utilisent plusieurs emplacements possibles pour les emplacements. Cela signifie qu'il existe toujours une capacité de secours si l'emplacement de placement préféré n'est pas disponible.

- Rendre la disponibilité des jeux plus résistante. Des pannes peuvent survenir. Avec une file d'attente multi-sites, un ralentissement ou une panne ne doit pas nécessairement affecter l'accès des joueurs à votre jeu.
- Utiliser plus efficacement la capacité de la flotte supplémentaire. Pour faire face à une augmentation imprévue de la demande des joueurs, les files d'attente permettent d'accéder rapidement à une capacité d'hébergement supplémentaire. Les emplacements des flottes dans une file d'attente fournissent une capacité de réserve les uns pour les autres. Les emplacements augmentent ou diminuent en fonction de la demande des joueurs.
- Obtenez des statistiques sur l'emplacement des sessions de jeu et les performances des files d'attente. Amazon GameLift diffuse des métriques de file d'attente, notamment des statistiques sur les succès et les échecs des placements, le nombre de demandes dans la file d'attente et le temps moyen que les demandes passent dans la file d'attente. Vous pouvez consulter ces statistiques dans la GameLift console Amazon ou dansCloudWatch.

Pour commencer à gérer les files d'attente, consultezConception d'une file d'attente de sessions de jeu.

Conception d'une file d'attente de sessions de jeu

Cette rubrique explique comment concevoir une file d'attente qui offre une expérience aux joueurs avec une latence minimale et qui utilise efficacement les ressources d'hébergement. Pour plus d'informations sur les files d'attente des sessions de jeu et leur fonctionnement, consultezConfiguration des GameLift files d'attente Amazon pour le placement des sessions de jeu.

Ces GameLift fonctionnalités Amazon nécessitent des files d'attente :

- Mise en relation avec FlexMatch
- Utiliser des instances Spot avec Amazon GameLift

Définissez l'étendue de votre file d'attente

La population de joueurs de votre jeu peut comprendre des groupes de joueurs qui ne devraient pas jouer ensemble. Par exemple, si vous publiez votre jeu en deux langues, chaque langue doit disposer de ses propres serveurs de jeu.

Pour configurer le placement des sessions de jeu en fonction de votre population de joueurs, créez une file d'attente distincte pour chaque segment de joueur. Délimitez chaque file d'attente pour placer

les joueurs sur les bons serveurs de jeu. Les méthodes les plus courantes pour délimiter les files d'attente sont les suivantes :

- Par zones géographiques. Lorsque vous déployez vos serveurs de jeu dans plusieurs zones géographiques, vous pouvez créer des files d'attente pour les joueurs à chaque endroit afin de réduire la latence des joueurs.
- Par des variantes de build ou de script. Si vous possédez plusieurs variantes de votre serveur de jeu, vous soutenez peut-être des groupes de joueurs qui ne peuvent pas jouer au cours des mêmes sessions de jeu. Par exemple, les versions ou les scripts des serveurs de jeu peuvent prendre en charge différentes langues ou différents types d'appareils.
- Par type d'événement. Vous pouvez créer une file d'attente spéciale pour gérer les jeux des participants à des tournois ou à d'autres événements spéciaux.

Création d'une politique de latence pour les joueurs

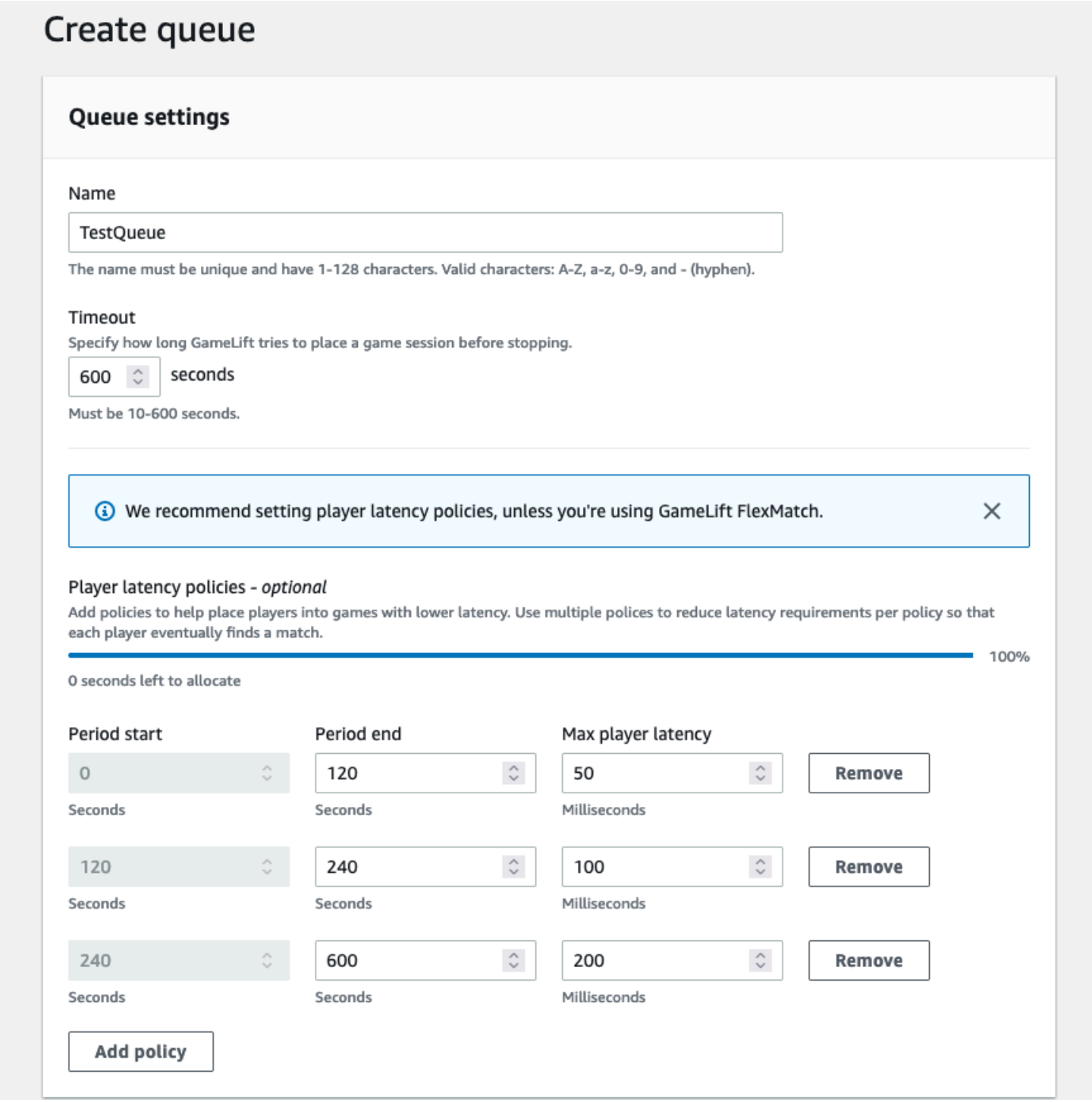
Si vos demandes de placement incluent des données sur la latence des joueurs, l'algorithme trouve les sessions de jeu aux endroits où la latence moyenne est la plus faible pour tous les joueurs. Le fait de placer des sessions de jeu en fonction de la latence moyenne des joueurs GameLift empêche Amazon de placer la plupart des joueurs dans des jeux à latence élevée. Cependant, Amazon place GameLift toujours les joueurs avec une latence extrême. Pour répondre aux besoins de ces joueurs, créez des politiques de latence pour les joueurs.

Une politique de latence pour les joueurs GameLift empêche Amazon de placer une session de jeu demandée à un endroit où les joueurs concernés par la demande connaîtraient une latence supérieure à la valeur maximale. Les politiques relatives à la latence des joueurs peuvent également GameLift empêcher Amazon de faire correspondre les demandes de session de jeu à des joueurs à latence plus élevée.

> **Tip**
>
> Pour gérer des règles spécifiques à la latence, telles que l'exigence d'une latence similaire pour tous les joueurs d'un groupe, vous pouvez utiliser Amazon GameLift FlexMatch pour créer des règles de matchmaking basées sur la latence.

Par exemple, considérez cette file d'attente avec un délai d'attente de 5 minutes et les politiques de latence des joueurs suivantes :

1. Passez 120 secondes à rechercher un endroit où toutes les latences des joueurs sont inférieures à 50 millisecondes.
2. Passez 120 secondes à rechercher un endroit où toutes les latences des joueurs sont inférieures à 100 millisecondes.
3. Passez le temps d'attente restant jusqu'à l'expiration du délai imparti à rechercher un endroit où toutes les latences des joueurs sont inférieures à 200 millisecondes.



Créez une file d'attente multi-sites

Nous recommandons une conception multi-emplacements pour toutes les files d'attente. Cette conception peut améliorer la vitesse de placement et la résilience de l'hébergement. Une conception multi-emplacements est requise pour utiliser les données de latence des joueurs afin de permettre aux joueurs de participer à des sessions de jeu avec une latence minimale. Si vous créez des files

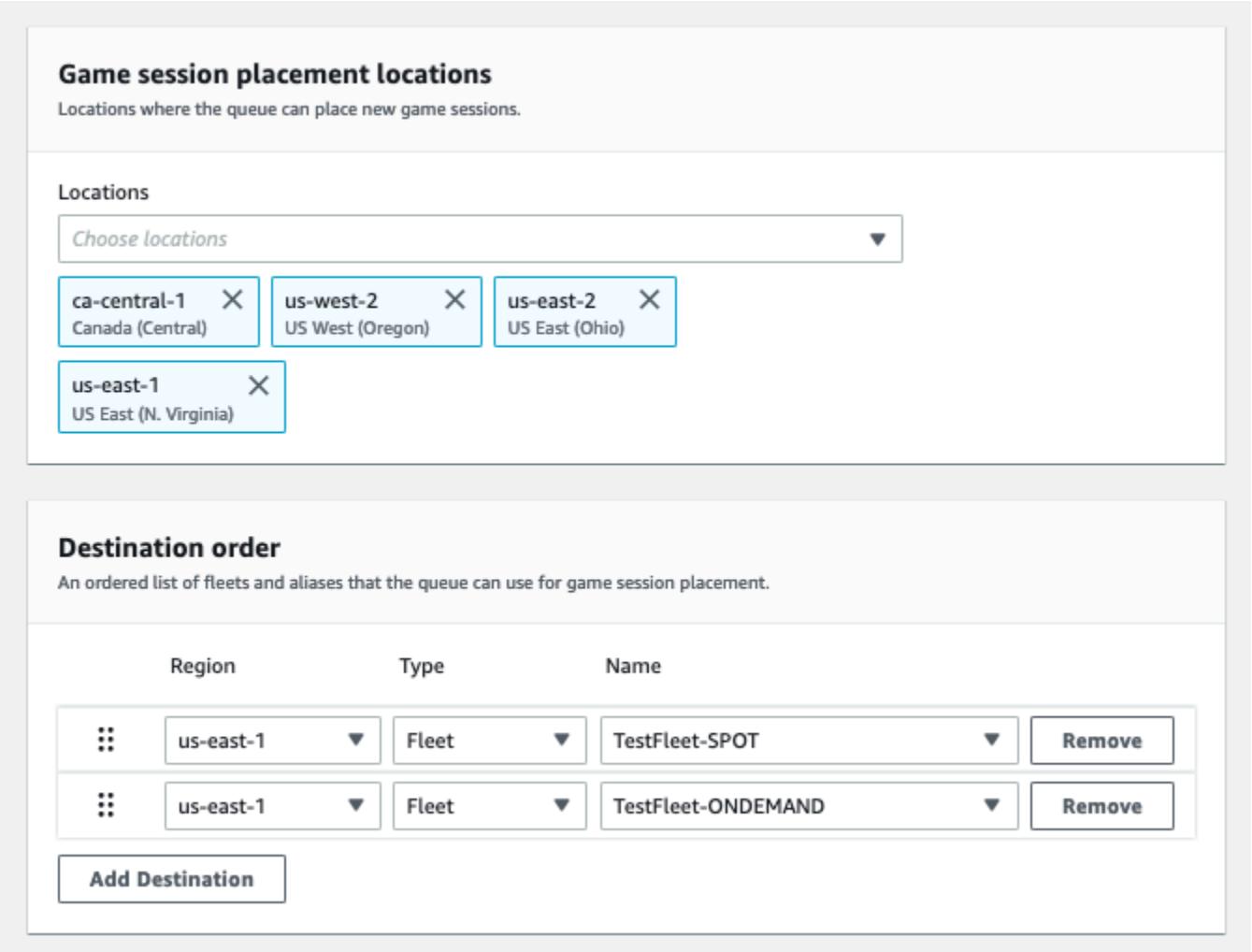
d'attente multi-sites qui utilisent des flottes d'instances Spot, suivez les instructions figurant dans. Tutoriel : Configuration d'une file d'attente de sessions de jeu pour les instances Spot

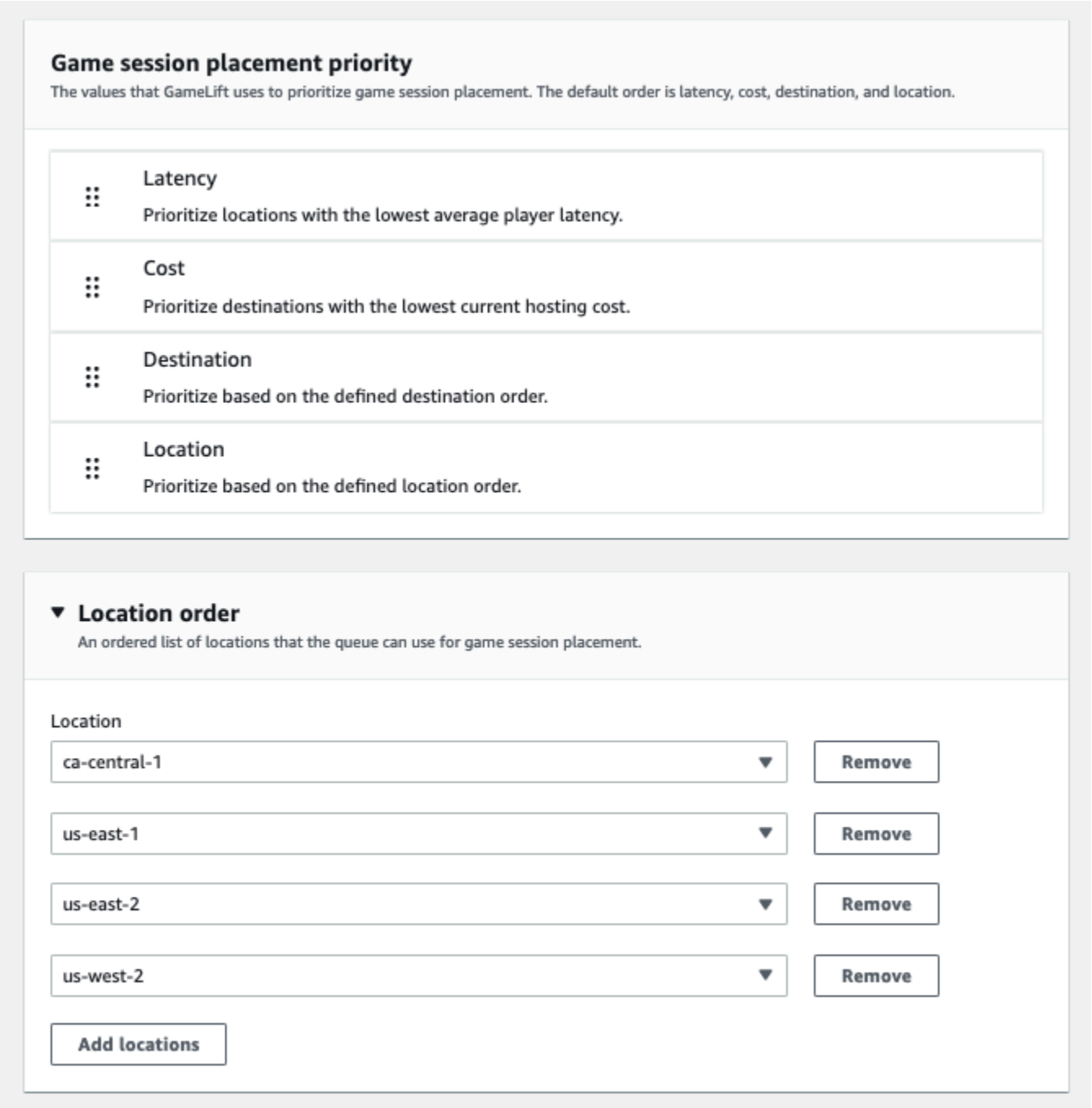
L'un des moyens de créer une file d'attente multi-sites consiste à ajouter un parc multi-sites à une file d'attente. Ainsi, la file d'attente peut placer des sessions de jeu dans n'importe quel emplacement de la flotte. Vous pouvez également ajouter d'autres flottes avec des configurations différentes ou des sites d'origine à des fins de redondance. Si vous utilisez un parc d'instances Spot multi-sites, suivez les meilleures pratiques et incluez un parc d'instances à la demande avec les mêmes emplacements.

L'exemple suivant décrit le processus de conception d'une file d'attente multi-emplacements de base. Dans cet exemple, nous utilisons deux flottes : une flotte d'instances Spot et une flotte d'instances à la demande. Chaque flotte possède les emplacements Régions AWS de placement suivants : `us-east-1`us-east-2,`ca-central-1`, et`us-west-2`.

Pour créer une file d'attente multi-sites de base avec des flottes multi-sites

1. Choisissez l'emplacement dans lequel créer la file d'attente. Vous pouvez minimiser la latence des demandes en plaçant la file d'attente à un endroit proche de l'endroit où vous avez déployé le service client. Dans cet exemple, nous créons la file d'attente dans`us-east-1`.
2. Créez une nouvelle file d'attente et ajoutez vos flottes multi-sites comme destinations de file d'attente. L'ordre de destination détermine la manière dont Amazon GameLift place les sessions de jeu. Dans cet exemple, nous listons le parc d'instances Spot en premier et le parc d'instances à la demande en second.
3. Définissez l'ordre de priorité de placement des sessions de jeu dans la file d'attente. Cet ordre détermine où la file d'attente recherche en premier lieu un serveur de jeu disponible. Dans cet exemple, nous utilisons l'ordre de priorité par défaut.
4. Définissez l'ordre de localisation. Si vous ne définissez pas l'ordre des lieux, Amazon GameLift utilise les lieux par ordre alphabétique.


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



Prioriser le placement des sessions de jeu

Amazon GameLift utilise l'algorithme FleetIQ pour déterminer où placer une nouvelle session de jeu en fonction d'un ensemble ordonné de critères. Vous pouvez utiliser l'ordre de priorité par défaut ou personnaliser l'ordre.

Ordre de priorité par défaut

Pour les demandes de placement qui incluent les données de latence des joueurs, FleetIQ hiérarchise les critères de placement des sessions de jeu dans l'ordre par défaut suivant :

1. Latence : latence moyenne la plus faible pour tous les joueurs concernés par la requête.
2. Coût : coût d'hébergement le plus bas, si la latence est égale sur plusieurs sites. Le coût d'hébergement est principalement basé sur une combinaison du type d'instance et de l'emplacement.
3. Destination : ordre de destination, si la latence et le coût sont égaux sur plusieurs sites. FleetIQ hiérarchise les destinations en fonction de l'ordre indiqué dans la configuration de la file d'attente.
4. Emplacement : ordre de localisation, si la latence, le coût et la destination sont identiques dans plusieurs emplacements. FleetIQ hiérarchise les emplacements en fonction de l'ordre indiqué dans la configuration de la file d'attente.

Ordre de priorité personnalisé

Pour personnaliser l'ordre de priorité d'une file d'attente dans la GameLiftconsole Amazon, faites glisser la valeur de priorité vers la position souhaitée. Pour personnaliser l'ordre de priorité d'une file d'attente à l'aide du AWS Command Line Interface (AWS CLI), utilisez la create-game-session-queuecommande avec l'--priority-configurationoption. Vous pouvez utiliser cette commande pour créer une nouvelle file d'attente ou pour mettre à jour une file d'attente existante.

L'algorithme FleetIQ ajoute tous les critères qui ne sont pas explicitement mentionnés à la fin de votre liste, selon l'ordre par défaut. Si vous incluez le critère de localisation dans votre configuration de priorité, vous devez également fournir une liste ordonnée d'emplacements.

Concevez plusieurs files d'attente selon vos besoins

En fonction de votre jeu et de vos joueurs, vous souhaiterez peut-être créer plusieurs files d'attente de sessions de jeu. Lorsque le service client de votre jeu demande une nouvelle session de jeu, il indique la file d'attente de sessions de jeu à utiliser. Pour vous aider à déterminer s'il convient d'utiliser plusieurs files d'attente, pensez à :

- Variantes de votre serveur de jeu. Vous pouvez créer une file d'attente distincte pour chaque variante de votre serveur de jeu. Toutes les flottes d'une file d'attente doivent déployer des serveurs de jeu compatibles. En effet, les joueurs qui utilisent la file d'attente pour rejoindre des parties doivent pouvoir jouer sur n'importe quel serveur de jeu de la file d'attente.
- Différents groupes de joueurs. Vous pouvez personnaliser la façon dont Amazon GameLift place les sessions de jeu en fonction du groupe de joueurs. Par exemple, vous pouvez avoir besoin de

files d'attente personnalisées pour certains modes de jeu qui nécessitent un type d'instance ou une configuration d'exécution spéciaux. Vous pouvez également avoir besoin d'une file d'attente spéciale pour gérer les emplacements pour un tournoi ou un autre événement.

- Statistiques de la file d'attente des sessions de jeu. Vous pouvez configurer des files d'attente en fonction de la manière dont vous souhaitez collecter les statistiques de placement des sessions de jeu. Pour plus d'informations, veuillez consulter GameLiftMétriques Amazon pour les files d'attente.

Évaluez les métriques de file d'

Utilisez les métriques pour évaluer les performances de vos files d'attente. Vous pouvez consulter les statistiques relatives aux files d'attente dans la GameLiftconsole Amazon ou sur AmazonCloudWatch. Pour obtenir une liste et une description des métriques de file d'attente, consultezGameLiftMétriques Amazon pour les files d'attente.

Les métriques de file d'attente peuvent fournir des informations sur les points suivants :

- Performances globales des files d'attente : les métriques des files d'attente indiquent dans quelle mesure une file d'attente répond aux demandes de placement. Ces indicateurs peuvent également vous aider à identifier quand et pourquoi les placements échouent. Pour les files d'attente dont les flottes sont redimensionnées manuellement, les `QueueDepth` métriques `AverageWaitTime` et peuvent indiquer à quel moment vous devez ajuster la capacité d'une file d'attente.
- Performances de l'algorithme FleetIQ : pour les demandes de placement utilisant l'algorithme FleetIQ, les métriques indiquent à quelle fréquence l'algorithme trouve le placement idéal pour les sessions de jeu. Le placement peut donner la priorité à l'utilisation des ressources présentant la latence la plus faible pour les joueurs ou des ressources les moins coûteuses. Il existe également des mesures d'erreur qui identifient les raisons courantes pour lesquelles Amazon ne GameLift parvient pas à trouver le placement idéal. Pour plus d'informations sur les métriques, consultez la section Surveillez Amazon GameLift avec Amazon CloudWatch.
- Placements spécifiques à un emplacement : pour les files d'attente multi-sites, les statistiques indiquent les placements réussis par emplacement. Pour les files d'attente qui utilisent l'algorithme FleetIQ, ces données fournissent des informations utiles sur l'endroit où se produit l'activité des joueurs.

Lorsque vous évaluez les mesures relatives aux performances de l'algorithme FleetIQ, tenez compte des conseils suivants :

- Pour suivre le taux de recherche du placement idéal dans la file d'attente, utilisez la `PlacementsSucceeded` métrique en combinaison avec les métriques FleetIQ pour obtenir la latence la plus faible et le prix le plus bas.
- Pour augmenter le taux de recherche de l'emplacement idéal d'une file d'attente, consultez les indicateurs d'erreur suivants :
  - S'il `FirstChoiceOutOfCapacity` est élevé, ajustez l'échelle de capacité pour les flottes de la file d'attente.
  - Si la métrique `FirstChoiceNotViable` d'erreur est élevée, examinez vos flottes d'instances Spot. Les flottes d'instances Spot sont considérées comme non viables lorsque le taux d'interruption d'un type d'instance particulier est trop élevé. Pour résoudre ce problème, modifiez la file d'attente afin d'utiliser des flottes d'instances Spot avec différents types d'instances. Nous vous recommandons d'inclure des flottes d'instances Spot avec différents types d'instances dans chaque emplacement.

Meilleures pratiques en matière de files d'attente pour les sessions de GameLift jeu sur Amazon

Voici quelques bonnes pratiques qui peuvent vous aider à créer des files d'attente de sessions de jeu efficaces pour le placement des sessions de jeu.

Meilleures pratiques pour les files d'attente, quel que soit le type de flotte

Une file d'attente contient la liste des destinations de la flotte où de nouvelles sessions de jeu peuvent être placées. Chaque parc peut disposer d'instances déployées dans plusieurs zones géographiques. Lorsque vous choisissez un emplacement, la file d'attente sélectionne une combinaison entre un parc et un emplacement du parc. Vous fournissez à la file d'attente un ensemble de priorités à utiliser lors du choix d'un emplacement.

Tenez compte des directives et des meilleures pratiques suivantes :

- Ajoutez des flottes à des endroits qui couvrent vos joueurs. Vous pouvez ajouter des flottes et des alias à n'importe quel emplacement disponible. La localisation est importante si vous effectuez des placements en fonction de la latence signalée par les joueurs.
- Utilisez des alias pour toutes les flottes. Attribuez un alias à chaque flotte d'une file d'attente et utilisez les noms d'alias lorsque vous définissez des destinations dans votre file d'attente.

- Utilisez une version de jeu ou un script identique ou similaire pour toutes les flottes. La file d'attente peut permettre aux joueurs de participer à des sessions de jeu sur n'importe quelle flotte de la file. Les joueurs doivent pouvoir participer à n'importe quelle session de jeu sur n'importe quelle flotte.
- Créez des flottes à au moins deux endroits. En hébergeant les serveurs de jeu dans au moins un autre emplacement, vous réduisez l'impact des pannes régionales sur vos joueurs. Vous pouvez réduire la taille de vos flottes de sauvegarde et utiliser la mise à l'échelle automatique pour augmenter la capacité si l'utilisation augmente.
- Donnez la priorité à l'emplacement de vos sessions de jeu. Une file d'attente hiérarchise les choix de placement en fonction de plusieurs éléments, notamment l'ordre de la liste de destinations.
- Créez votre file d'attente au même endroit que votre service client. En plaçant votre file d'attente à proximité de votre service client, vous pouvez minimiser la latence des communications.
- Utilisez des flottes avec plusieurs sites. Utilisez la configuration du filtre de file d'attente pour empêcher la file d'attente de placer des sessions de jeu à des emplacements spécifiques. Vous pouvez utiliser au moins deux flottes multi-sites avec des sites d'attache différents pour atténuer l'impact du placement des parties lors d'une panne régionale.
- Utilisez le même paramètre de certificat TLS pour toutes les flottes. Les clients de jeu qui se connectent aux sessions de jeu de vos flottes doivent disposer de protocoles de communication compatibles.

Meilleures pratiques en matière de files d'attente avec les flottes Spot

Si votre file d'attente inclut des flottes Spot, configurez une file résiliente. Cela permet de tirer parti des économies réalisées grâce aux flottes Spot tout en minimisant l'effet des interruptions de session de jeu. Pour obtenir de l'aide sur la création correcte de flottes et de files d'attente de sessions de jeu à utiliser avec les flottes Spot, voir. Tutoriel : Configuration d'une file d'attente de sessions de jeu pour les instances Spot Pour plus d'informations sur les instances Spot, consultezUtiliser des instances Spot avec Amazon GameLift.

Outre les meilleures pratiques générales décrites dans la section précédente, prenez en compte les meilleures pratiques spécifiques à Spot :

- Créez au moins une flotte à la demande dans chaque site. Les flottes à la demande fournissent des serveurs de jeu de secours à vos joueurs. Vous pouvez réduire la taille de vos flottes de sauvegarde jusqu'à ce que vous en ayez besoin et utiliser la mise à l'échelle automatique pour augmenter la capacité à la demande lorsque les flottes Spot ne sont pas disponibles.

- Sélectionnez différents types d'instances au sein de plusieurs flottes Spot au même endroit. Si un type d'instance Spot devient temporairement indisponible, l'interruption ne concerne qu'un seul parc d'instances Spot sur place. La meilleure pratique consiste à choisir des types d'instances largement disponibles et à utiliser des types d'instances de la même famille (par exemple, m5.large, m5.xlarge, m5.2xlarge). Utilisez la GameLiftconsole Amazon pour consulter les données de tarification historiques pour les types d'instances.

Création d'une file d'attente de sessions de jeu

Des files d'attente sont utilisées pour placer de nouvelles sessions de jeu avec les meilleures ressources d'hébergement disponibles dans plusieurs flottes et régions. Pour en savoir plus sur la création de files d'attente pour votre jeu, consultez Conception d'une file d'attente de sessions de jeu.

Dans un client de jeu, les nouvelles sessions de jeu sont démarrées avec des files d'attente en utilisant des demandes de placement. Pour en savoir plus sur le placement des sessions de jeu, consultezCréez des sessions de jeu.

Lors de la mise à jour de la destination d'une file d'attente, il existe une courte période de transition (jusqu'à 30 secondes) pendant laquelle les sessions de jeu placées sur les destinations de la file d'attente peuvent toujours se terminer sur l'ancienne flotte.

Console

1. Dans la GameLiftconsole Amazon, sur la page de navigation, choisissez Files d'attente.
2. Sur la page Queues (Files d'attente), choisissez Create queue (Créer une nouvelle file d'attente).
3. Sur la page Créer une file d'attente, sous Paramètres de file d'attente, procédez comme suit :

    a. Dans Nom, entrez un nom de file d'attente.

    b. Pour Timeout, saisissez la durée pendant laquelle vous souhaitez qu'Amazon GameLift essaie de placer une session de jeu avant de l'arrêter. Amazon GameLift recherche les ressources disponibles sur n'importe quel parc jusqu'à ce que la demande expire.

    c. (Facultatif) Pour les politiques de latence des joueurs, saisissez la durée pendant laquelle Amazon GameLift doit rechercher des ressources dans les limites de la latence maximale définie. Ajoutez des politiques supplémentaires pour réduire progressivement la latence maximale. Pour ajouter des politiques supplémentaires, choisissez Ajouter une politique.

4. Sous Lieux de placement des sessions de jeu, sélectionnez les emplacements à inclure dans la file d'attente. Par défaut, tous les emplacements sont inclus. Toutes les flottes de la file d'attente doivent avoir la même configuration de certificat. Toutes les flottes doivent exécuter des versions de jeu compatibles avec les clients du jeu utilisant la file d'attente.
5. Sous Ordre des destinations, ajoutez une ou plusieurs destinations à la file d'attente.
   a. Choisissez Add destination (Ajouter une destination).
   b. Sélectionnez l'emplacement dans lequel se trouve la destination.
   c. Sélectionnez le type correspondant à votre destination.
   d. Dans la liste des noms de flottes et d'alias obtenue, sélectionnez celui que vous souhaitez ajouter.
   e. Si vous avez plusieurs destinations, définissez l'ordre par défaut en faisant glisser l'icône à six points vers la gauche de la destination. Amazon GameLift utilise cette commande lors de la recherche de destinations pour les ressources disponibles afin d'ouvrir une nouvelle session de jeu.
6. Pour la priorité de placement des sessions de jeu, ajoutez et faites glisser les valeurs de latence, de coût, de destination et de localisation pour définir la manière dont Amazon GameLift hiérarchise les flottes dans votre file d'attente. Pour plus d'informations sur la hiérarchisation des flottes, consultez. Prioriser le placement des sessions de jeu
7. Ajoutez des lieux à votre ordre de localisation et faites-les glisser vers la priorité que la file d'attente doit utiliser. Si la localisation est la dernière priorité pour le placement des sessions de jeu, Amazon l'GameLiftutilise comme point décisif.
8. (Facultatif) Sous Paramètres de notification d'événements, procédez comme suit :
   a. Sélectionnez ou créez une rubrique SNS pour recevoir des notifications d'événements liés au placement. Pour plus d'informations sur les notifications d'événements, consultezConfigurer une notification d'événement pour le placement des sessions de jeu.
   b. Ajoutez des données d'événement personnalisées à ajouter aux événements créés par cette file d'attente.
9. (Facultatif) Ajoutez des balises. Pour plus d'informations sur le balisage, consultez la section Ressources sur le balisage. AWS
10. Sélectionnez Create (Créer).

AWS CLI

Example Créer une file d'attente

L'exemple suivant crée une file d'attente de sessions de jeu avec les configurations suivantes :

- Un délai d'attente de cinq minutes
- Deux destinations pour les flottes
- Filtres pour n'autoriser que les emplacements situés dans leus-east-1,us-east-2. us-west-2, et ca-central-1
- Priorise les destinations en fonction du coût, puis des lieux dans l'ordre défini.

```
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**
>
> Vous pouvez obtenir les valeurs ARN du parc et des alias en appelant l'un ou l'autre describe-fleet-attributesou en appelant describe-alias avec l'ID du parc ou de l'alias.

Si la `create-game-session-queue` demande aboutit, Amazon GameLift renvoie un GameSessionQueueobjet avec la nouvelle configuration de file d'attente. Vous pouvez désormais soumettre des demandes à la file d'attente en utilisant StartGameSessionPlacement.

Example Créez une file d'attente avec les politiques de latence des joueurs

L'exemple suivant crée une file d'attente de sessions de jeu avec les configurations suivantes :

- Un délai d'attente de dix minutes

- Trois destinations pour les flottes
- Un ensemble de politiques de latence pour les joueurs

```
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" \
```

Si la `create-game-session-queue` demande aboutit, Amazon GameLift renvoie un GameSessionQueueobjet avec la nouvelle configuration de file d'attente.

Configurer une notification d'événement pour le placement des sessions de jeu

Vous pouvez utiliser les notifications d'événements pour suivre l'état des demandes de placement individuelles. Nous vous recommandons de configurer des notifications d'événements pour tous les jeux dont l'activité de placement est importante.

Il existe deux options pour configurer les notifications d'événement.

- Demandez à Amazon de GameLift publier des notifications d'événements dans une rubrique Amazon Simple Notification Service (Amazon SNS) à l'aide d'une file d'attente.
- Utilisez les EventBridge événements Amazon publiés automatiquement et sa suite d'outils pour gérer les événements.

Pour obtenir la liste des événements de placement de session de jeu émis par AmazonGameLift, consultezÉvénements de placement de sessions de jeu.

Configurer une rubrique SNS

Pour GameLift qu'Amazon publie tous les événements générés par une file d'attente de sessions de jeu dans un sujet, définissez un sujet dans le champ cible des notifications.

Pour configurer une rubrique SNS pour la notification d'GameLiftévénements Amazon

1. Connectez-vous à AWS Management Console et ouvrez la console Amazon SNS à l'adresse https://console.aws.amazon.com/sns/v3/home.
2. Sur la page Sujets SNS, choisissez Créer un sujet et suivez les instructions pour créer votre sujet.
3. Dans la section Politique d'accès, procédez comme suit :

   a. Choisissez la méthode avancée.

   b. Ajoutez la section en gras suivante de l'objet JSON à la politique existante.

```
{
  "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. (Facultatif) Ajoutez un contrôle d'accès supplémentaire à la rubrique en ajoutant des conditions à la politique de ressources.

4. Choisissez Create topic (Créer une rubrique).
5. Après avoir créé votre rubrique SNS, ajoutez-la aux files d'attente lors de la création de la file d'attente ou modifiez une file d'attente existante pour l'ajouter.

Configuration d'une rubrique SNS avec chiffrement côté serveur

Le chiffrement côté serveur (SSE) vous permet de stocker des données sensibles dans des rubriques cryptées. SSE protège le contenu des messages dans les rubriques Amazon SNS à l'aide de clés gérées dans AWS Key Management Service (AWS KMS). Pour plus d'informations sur le chiffrement côté serveur avec Amazon SNS, consultez la section Chiffrement au repos dans le guide du développeur Amazon Simple Notification Service.

Pour configurer une rubrique SNS avec chiffrement côté serveur, consultez les rubriques suivantes :

- Création d'une clé dans le Guide du AWS Key Management Service développeur
- Activation de SSE pour une rubrique dans le guide du développeur d'Amazon Simple Notification Service

Lorsque vous créez votre clé KMS, utilisez la politique de clé KMS suivante :

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

Configurer EventBridge

Amazon publie GameLift automatiquement tous les événements de placement de sessions de jeu surEventBridge. Avec, EventBridge vous pouvez définir des règles pour que les événements soient acheminés vers des cibles pour traitement. Par exemple, vous pouvez définir une règle pour acheminer l'événement `PlacementFulfilled` vers une AWS Lambda fonction qui gère les tâches qui précèdent la connexion à une session de jeu. Pour en savoir plusEventBridge, consultez Qu'est-ce qu'Amazon EventBridge ? dans le guide de EventBridge l'utilisateur Amazon.

Voici quelques exemples de EventBridge règles à utiliser avec les GameLift files d'attente Amazon :

Fait correspondre les événements de toutes les GameLift files d'attente Amazon

```
{
    "source": [
        "aws.gamelift"
    ],
    "detail-type": [
```

```
        "GameLift Queue Placement Event"
    ]
}
```

Fait correspondre les événements d'une file d'attente spécifique

```
{
    "source": [
        "aws.gamelift"
    ],
    "detail-type": [
        "GameLift Queue Placement Event"
    ],
    "resources": [
        "arn:aws:gamelift:your_region:your_account:gamesessionqueue/your_queue_name"
    ]
}
```

Tutoriel : Configuration d'une file d'attente de sessions de jeu pour les instances Spot

Introduction

Ce didacticiel explique comment configurer le placement des sessions de jeu pour les jeux déployés sur des flottes Spot à faible coût. Les flottes Spot nécessitent des mesures supplémentaires pour maintenir la disponibilité continue des serveurs de jeu pour vos joueurs.

Public visé

Ce didacticiel s'adresse aux développeurs de jeux qui souhaitent utiliser des flottes Spot pour héberger des serveurs de jeu personnalisés ou des serveurs en temps réel.

Ce que tu vas apprendre

- Définissez le groupe de joueurs auquel s'adresse votre file d'attente de sessions de jeu.
- Créez une infrastructure de flotte adaptée à l'étendue de la file d'attente des sessions de jeu.
- Attribuez un alias à chaque flotte pour extraire l'ID de la flotte.
- Créez une file d'attente, ajoutez des flottes et hiérarchisez les GameLift endroits où Amazon place les sessions de jeu.
- Ajoutez des politiques de latence pour les joueurs afin de minimiser les problèmes de latence.

Prérequis

Avant de créer des flottes et des files d'attente pour le placement des sessions de jeu, effectuez les tâches suivantes :

- Vérifiez Comment GameLift fonctionne Amazon.
- Intégrez votre serveur de jeu à Amazon GameLift.
- Importez la version de votre serveur de jeu ou votre script en temps réel sur AmazonGameLift.
- Planifiez la configuration de votre flotte.

Étape 1 : Définissez l'étendue de votre file d'attente

Dans ce didacticiel, nous concevons une file d'attente pour un jeu qui possède une variante de build de serveur de jeu. À sa sortie, le jeu sera disponible dans deux destinations : Asie-Pacifique (Séoul) et Asie-Pacifique (Singapour). Comme ces sites sont proches les uns des autres, la latence n'est pas un problème pour nos joueurs.

Dans cet exemple, il existe un segment de joueurs, ce qui signifie que nous créons une file d'attente. À l'avenir, lorsque nous sortirons le jeu en Amérique du Nord, nous pourrons créer une deuxième file d'attente réservée aux joueurs nord-américains.

Pour plus d'informations, veuillez consulter Définissez l'étendue de votre file d'attente.

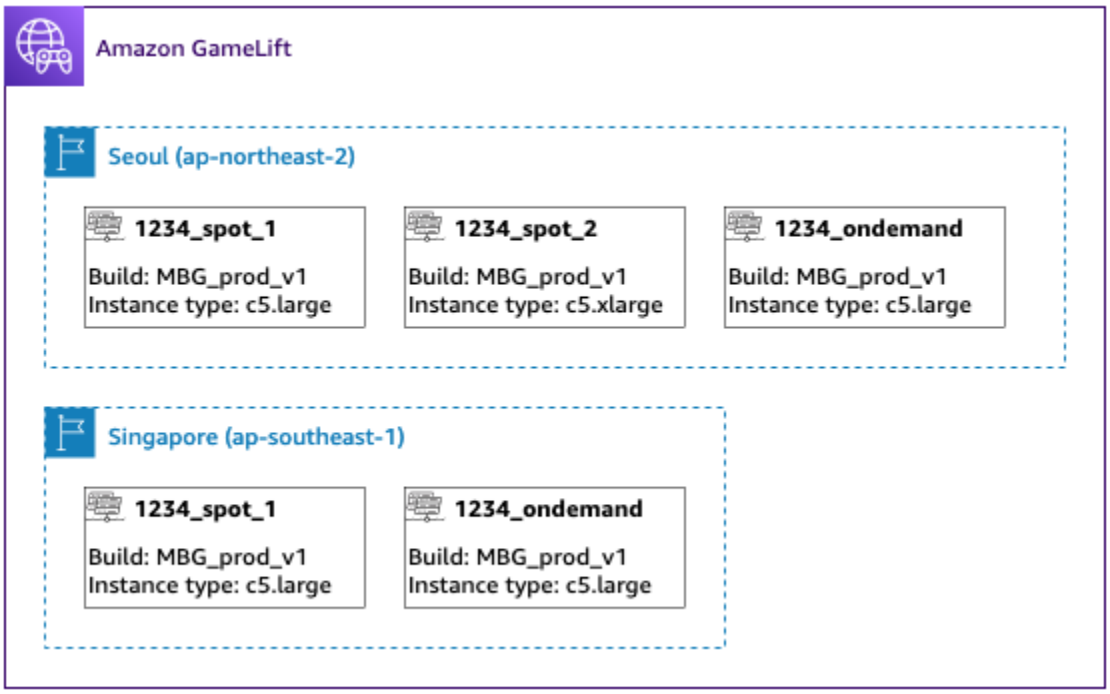
Étape 2 : Création d'une infrastructure de flotte Spot

Créez des flottes dans des lieux et à l'aide de versions de serveurs de jeu ou de scripts qui correspondent au périmètre que vous avez défini. Étape 1 : Définissez l'étendue de votre file d'attente

Dans ce didacticiel, nous allons créer une infrastructure à deux sites avec au moins un parc Spot et un parc On-Demand dans chaque site. Chaque flotte déploie la même version de serveur de jeu. De plus, nous prévoyons que le trafic de joueurs sera plus important sur le site de Séoul, c'est pourquoi nous y ajoutons davantage de flottes Spot.

Le schéma suivant montre l'exemple d'infrastructure de flotte Spot, avec 3 flottes sur le site ap-northeast-2 (Séoul) et 2 flottes sur le site ap-southeast-1 (Singapour). Toutes les instances des deux flottes utilisent la version MBG_Prod_v1. La flotte d'ap-northeast-2 contient les configurations de flotte suivantes : fleet 1234_spot_1 avec un type d'instance c5.large, fleet 1234_spot_2 avec un type d'instance c5.xlarge et fleet 1234_ondemand avec un type d'instance c5.large. La flotte d'ap-

southeast-1 contient les configurations de flotte suivantes : fleet 1234_spot_1 avec un type d'instance c5.large et fleet 1234_ondemand avec un type d'instance c5.large.

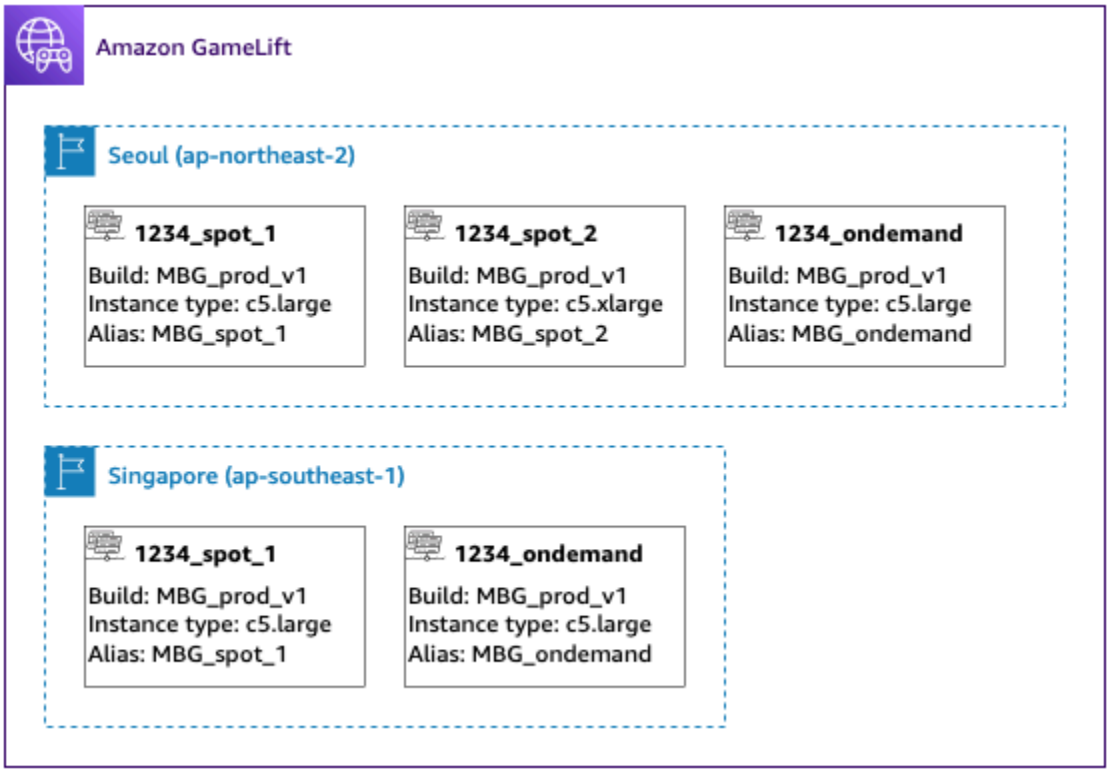


Étape 3 : Attribuer des alias à chaque flotte

Créez un nouvel alias pour chaque flotte de votre infrastructure. Les alias font abstraction des identités de flotte, ce qui rend le remplacement périodique du parc efficace. Pour plus d'informations sur la création d'alias, consultezAjouter un alias à une GameLift flotte Amazon.

Notre infrastructure de flotte compte cinq flottes. Nous créons donc cinq alias à l'aide de la stratégie de routage. Nous avons besoin de trois alias dans la zone Asie-Pacifique (Séoul) et de deux alias dans la zone Asie-Pacifique (Singapour).

Le schéma suivant montre l'infrastructure du parc Spot décrite à l'étape 2 avec des alias ajoutés à chaque parc. Fleet 1234_spot_1 possède l'alias MBG_Spot_1, Fleet 1234_spot_2 possède l'alias MBG_Spot_2 et Fleet 1234_ondemand possède l'alias MBG_OnDemand.



Pour plus d'informations, veuillez consulter Créez une file d'attente multi-sites.

Étape 4 : créer une file d'attente avec des destinations

Créez la file d'attente des sessions de jeu et ajoutez les destinations de votre flotte. Pour plus d'informations sur la création d'une file d'attente, consultezCréation d'une file d'attente de sessions de jeu.

Lors de la création de votre file d'attente :

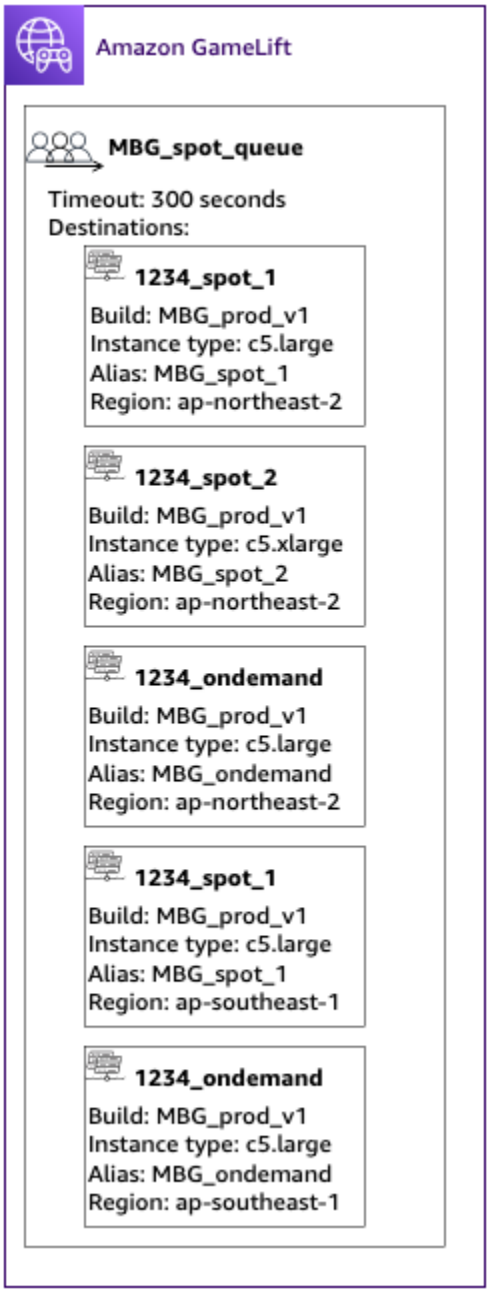
- Définissez le délai d'expiration par défaut sur 10 minutes. Plus tard, vous pourrez tester l'impact du délai d'attente sur le temps d'attente de vos joueurs pour accéder aux parties.
- Ignorez la section sur les politiques de latence des joueurs pour le moment. Nous aborderons ce point à l'étape suivante.
- Priorisez les flottes de votre file d'attente. Lorsque vous travaillez avec des flottes Spot, nous vous recommandons l'une des approches suivantes :
  - Si votre infrastructure utilise un emplacement principal avec des flottes dans un second emplacement à des fins de sauvegarde, hiérarchisez les flottes d'abord par emplacement, puis par type de flotte.

- Si votre infrastructure utilise plusieurs sites de la même manière, hiérarchisez les flottes par type de flotte, en plaçant les flottes Spot en haut de la file d'attente.

Pour ce didacticiel, nous créons une nouvelle file d'attente avec **`MBG_spot_queue`** le nom et ajoutons les alias de nos cinq flottes. Nous hiérarchisons ensuite les placements d'abord par emplacement, puis par type de flotte.

Sur la base de cette configuration, cette file d'attente tente toujours de placer de nouvelles sessions de jeu dans une flotte Spot à Séoul. Lorsque ces flottes sont pleines, la file d'attente utilise la capacité disponible sur la flotte Seoul On-Demand comme réserve. Si les trois flottes de Séoul ne sont pas disponibles, Amazon GameLift place des sessions de jeu sur les flottes de Singapour.

Le schéma suivant montre une file d'attente avec un délai d'attente de 300 secondes et des destinations prioritaires. Les destinations sont classées dans l'ordre suivant : 1234_spot_1 dans ap-northeast-2, 1234_spot_2 dans ap-northeast-2, 1234_ondemand dans ap-northeast-2, 1234_spot_1 dans ap-southeast-1 et 1234_ondemand dans ap-southeast-1.



Étape 5 : ajouter des limites de latence à la file d'attente

Notre jeu inclut des informations de latence dans les demandes de placement de session de jeu. Nous avons également une fonctionnalité de groupe de joueurs qui crée une session de jeu pour un groupe de joueurs. Nous pouvons inciter les joueurs à attendre un peu plus longtemps pour accéder à des jeux offrant l'expérience de jeu idéale. Nos tests de jeu montrent les observations suivantes :

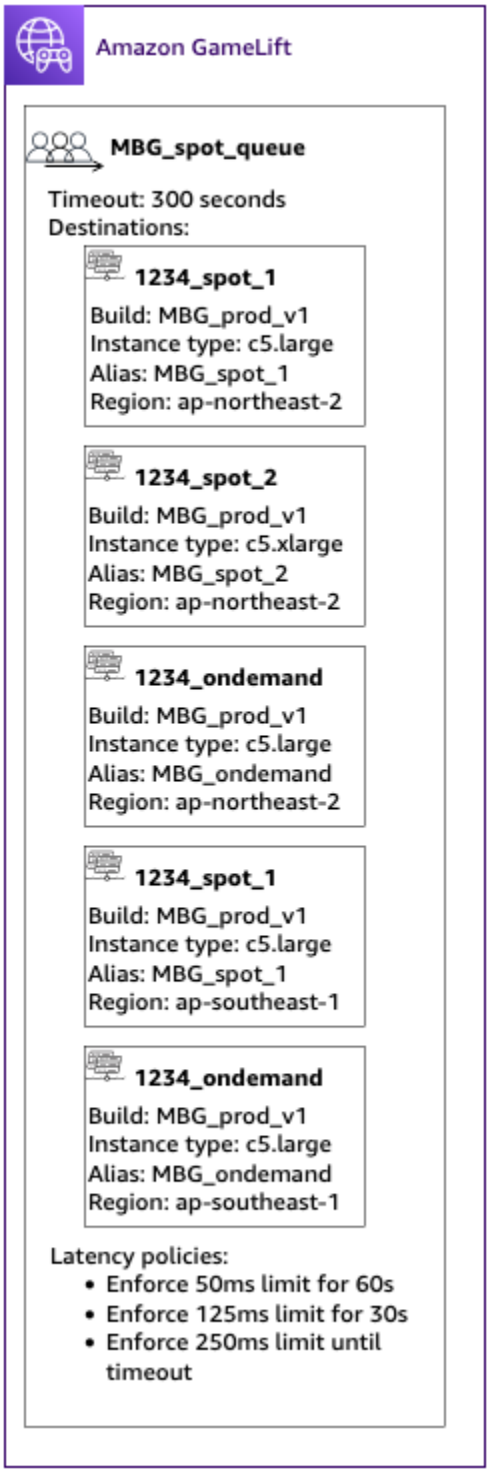
- Une latence inférieure à 50 millisecondes est idéale.

- Le jeu est injouable à des latences supérieures à 250 millisecondes.
- Les joueurs s'impatientent au bout d'une minute environ.

Pour notre file d'attente, avec un délai d'attente de 300 secondes, nous ajoutons des déclarations de politique limitant la latence autorisée. Les déclarations de politique autorisent progressivement des valeurs de latence plus élevées jusqu'à 250 millisecondes.

Avec cette politique, notre file d'attente recherche les emplacements présentant une latence idéale (moins de 50 millisecondes) pendant la première minute, puis assouplit la limite. La file d'attente n'inclut pas les emplacements où la latence du joueur est de 250 millisecondes ou plus.

Le schéma suivant montre la file d'attente à partir de la quatrième étape avec les politiques de latence des joueurs ajoutées. Les politiques de latence des joueurs stipulent qu'il faut appliquer une limite de 50 ms pendant 60 secondes, une limite de 125 ms pendant 30 secondes et une limite de 250 ms jusqu'à l'expiration du délai imparti.



Récapitulatif

Félicitations ! Voici ce que vous avez accompli :

- Vous disposez d'une file d'attente de sessions de jeu limitée à un segment de votre population de joueurs.
- Votre file d'attente utilise efficacement les flottes Spot et résiste aux interruptions Spot.
- Votre file d'attente donne la priorité aux flottes pour une expérience de jeu optimale.
- La file d'attente comporte des limites de latence pour protéger les joueurs contre les mauvaises expériences de jeu.

Vous pouvez désormais utiliser la file d'attente pour placer des sessions de jeu pour les joueurs qu'elle dessert. Lorsque vous faites des demandes de placement de session de jeu pour ces joueurs, faites référence au nom de cette file d'attente de session de jeu dans la demande. Pour plus d'informations sur les demandes de placement de sessions de jeuCréez des sessions de jeu, consultez ouIntégration d'un client de jeu pour les serveurs en temps réel.

Prochaines étapes :

- Créez votre propre file d'attente.
- Créez une file d'attente.
- Utilisez une file d'attente avec votre client de jeu.

Gérez les ressources à l'aide AWS CloudFormation

Vous pouvez l'utiliser AWS CloudFormation pour gérer vos GameLift ressources Amazon. Dans AWS CloudFormation, vous créez un modèle qui modélise chaque ressource, puis vous utilisez le modèle pour créer vos ressources. Pour mettre à jour des ressources, vous apportez des modifications à votre modèle et utilisez AWS CloudFormation pour mettre en œuvre les mises à jour. Vous pouvez organiser vos ressources en groupes logiques, appelés piles et ensembles de piles.

L'utilisation AWS CloudFormation pour gérer vos ressources GameLift d'hébergement Amazon constitue un moyen plus efficace de gérer des ensembles de AWS ressources. Vous pouvez utiliser le contrôle de version pour suivre les modifications apportées au modèle au fil du temps et coordonner les mises à jour effectuées par plusieurs membres de l'équipe. Vous pouvez également réutiliser des modèles. Par exemple, lors du déploiement d'un jeu dans plusieurs régions, vous pouvez utiliser le même modèle pour créer des ressources identiques dans chaque région. Vous pouvez également utiliser ces modèles pour déployer les mêmes ensembles de ressources dans une autre partition.

Pour plus d'informations sur AWS CloudFormation, consultez le AWS CloudFormationGuide de l'utilisateur . Pour consulter les informations relatives aux modèles de GameLift ressources Amazon, consultez la référence GameLift des types de ressources Amazon.

Bonnes pratiques

Pour obtenir des instructions détaillées sur l'utilisationAWS CloudFormation, consultez les AWS CloudFormationmeilleures pratiques dans le Guide de AWS CloudFormation l'utilisateur. En outre, ces bonnes pratiques sont particulièrement pertinentes pour AmazonGameLift.

- Gérez vos ressources de manière cohérente viaAWS CloudFormation. Si vous modifiez vos ressources, celles-ci AWS CloudFormation ne seront pas synchronisées avec vos modèles de ressources.
- Utilisez des piles et des ensembles de piles AWS CloudFormation pour gérer efficacement plusieurs ressources.
  - Utilisez des piles pour gérer des groupes de ressources connectées. Par exemple, une pile qui contient une version, une flotte qui fait référence à la version et un alias qui fait référence à la flotte. Si vous mettez à jour votre modèle pour remplacer une version, AWS CloudFormation remplace les flottes connectées à la version. AWS CloudFormationmet ensuite à jour les alias existants pour pointer vers les nouvelles flottes. Pour plus d'informations, consultez la section Utilisation des piles dans le Guide de l'AWS CloudFormationutilisateur.
  - Utilisez des ensembles de AWS CloudFormation piles si vous déployez des piles identiques sur plusieurs régions ou AWS comptes. Pour plus d'informations, consultez la section Utilisation des ensembles de piles dans le Guide de AWS CloudFormation l'utilisateur.
- Si vous utilisez des instances Spot, incluez une flotte à la demande comme sauvegarde. Nous vous recommandons de configurer vos modèles avec deux flottes dans chaque région, une flotte avec des instances Spot et une flotte avec des instances à la demande.
- Regroupez vos ressources spécifiques à un site et vos ressources globales dans des piles distinctes lorsque vous gérez des ressources sur plusieurs sites.
- Placez vos ressources globales à proximité des services qui les utilisent. Les ressources telles que les files d'attente et les configurations de matchmaking ont tendance à recevoir un volume élevé de demandes provenant de sources spécifiques. En plaçant vos ressources à proximité de la source de ces demandes, vous réduisez le temps de trajet des demandes et pouvez améliorer les performances globales.
- Placez votre configuration de mise en relation dans la même région que la file d'attente de session de jeu qu'elle utilise.

- Créez un alias distinct pour chaque flotte de la pile.

Utiliser des AWS CloudFormation piles

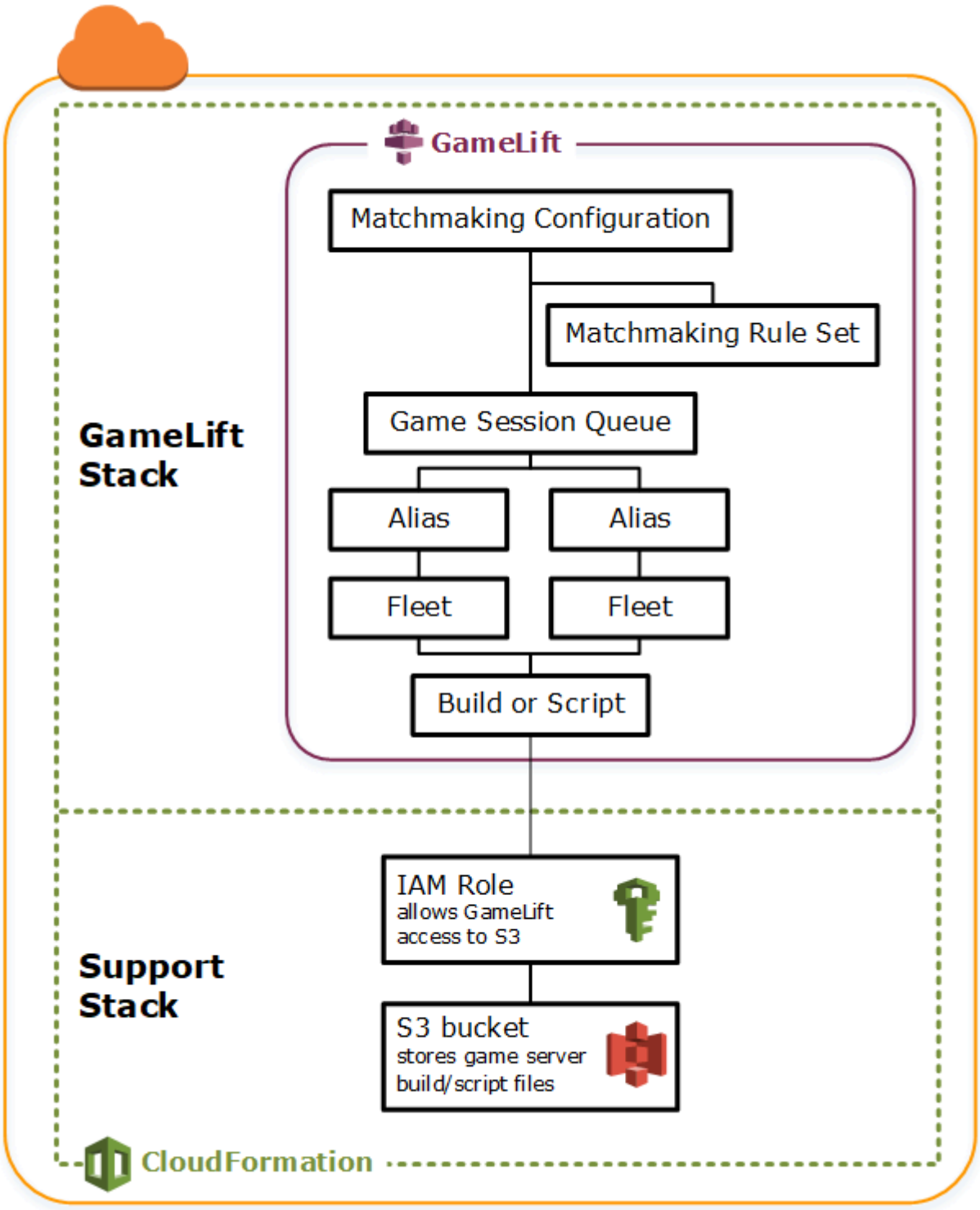
Nous vous recommandons d'utiliser les structures suivantes lors de la configuration de AWS CloudFormation piles pour les GameLift ressources Amazon. La structure optimale de votre stack varie selon que vous déployez votre jeu à un seul endroit ou à plusieurs endroits.

Des piles pour un seul emplacement

Pour gérer les GameLift ressources Amazon en un seul endroit, nous recommandons une structure à deux niveaux :

- Pile de support : cette pile contient des ressources dont dépendent vos GameLift ressources Amazon. Au minimum, cette pile doit inclure le compartiment S3 dans lequel vous stockez votre serveur de jeu personnalisé ou vos fichiers de script Realtime. La pile doit également inclure un rôle IAM qui GameLift autorise Amazon à récupérer vos fichiers depuis le compartiment S3 lors de la création d'une ressource de GameLift build ou de script Amazon. Cette pile peut également contenir d'autres AWS ressources utilisées avec votre jeu, telles que des tables DynamoDB, des clusters Amazon Redshift et des fonctions Lambda.
- GameLiftSuite Amazon : cette pile contient toutes vos GameLift ressources Amazon, y compris le build ou le script, un ensemble de flottes, des alias et la file d'attente des sessions de jeu. AWS CloudFormationcrée une ressource de build ou de script avec des fichiers stockés dans l'emplacement du bucket S3 et déploie le build ou le script sur une ou plusieurs ressources du parc. Chaque flotte doit avoir un alias correspondant. La file d'attente de session de jeu fait référence à tout ou partie des alias de flotte. Si vous l'utilisez FlexMatch pour le matchmaking, cette pile contient également une configuration de matchmaking et un ensemble de règles.

Le diagramme ci-dessous illustre une structure à deux piles pour le déploiement de ressources dans une seule région AWS.



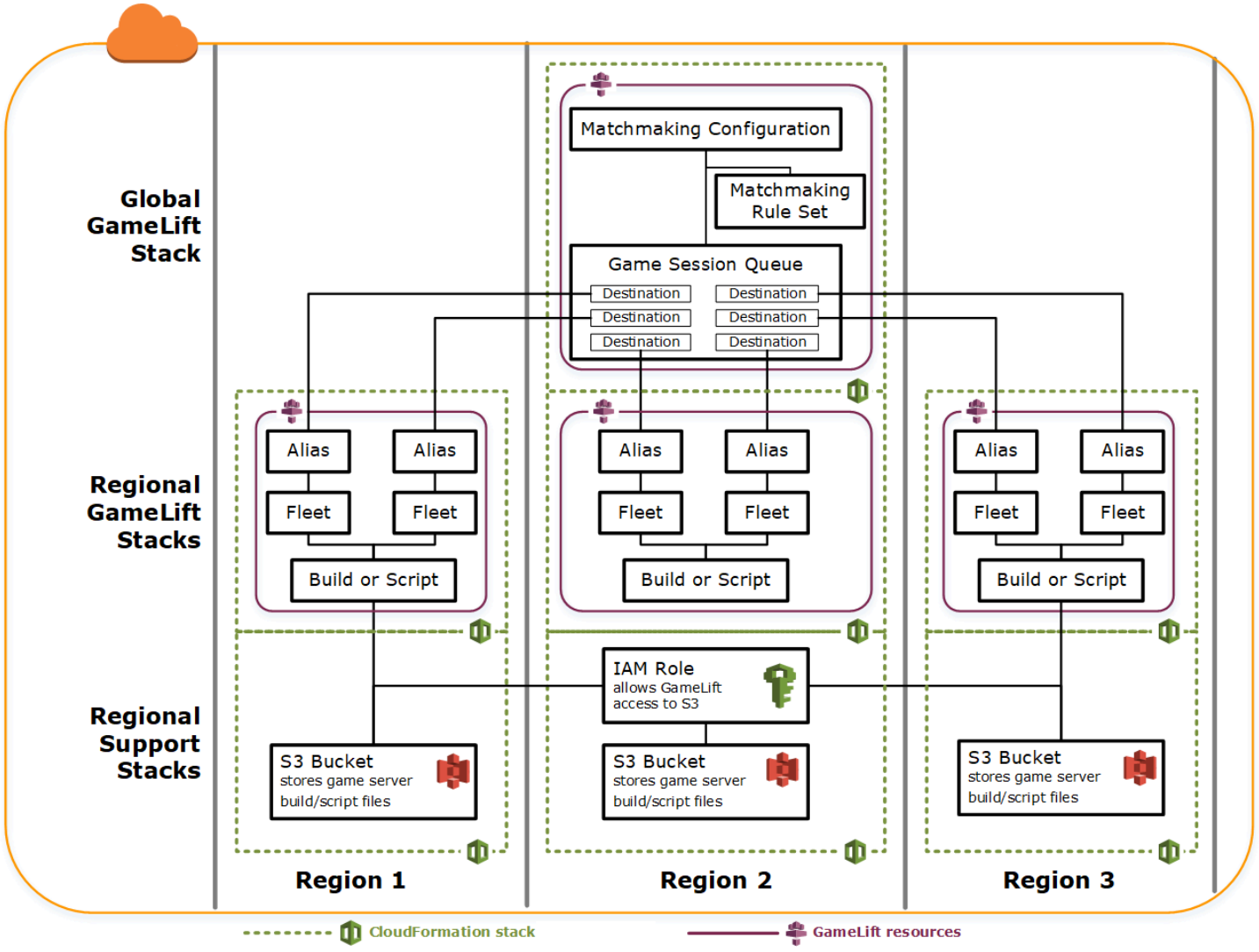
Des piles pour plusieurs régions

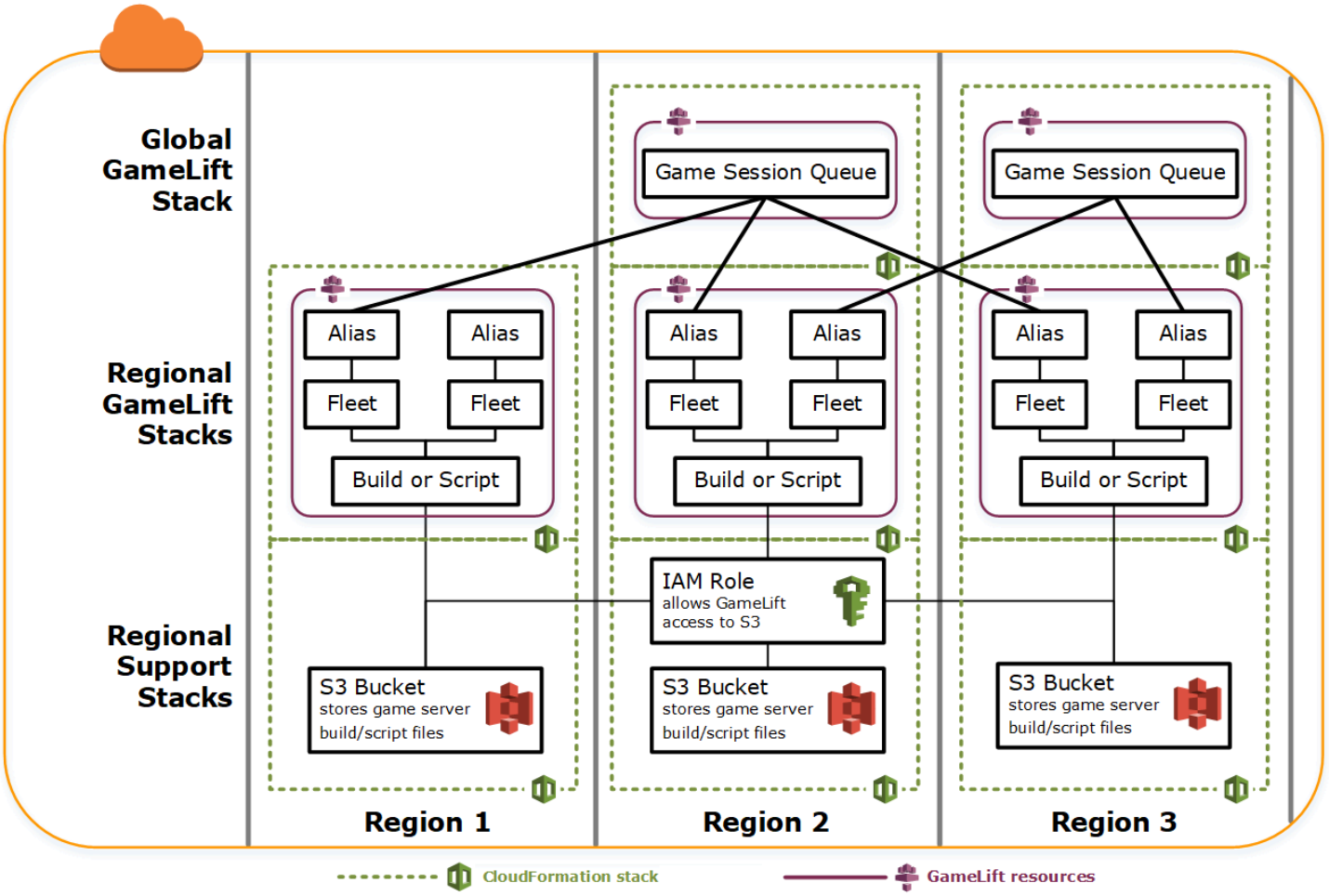
Lorsque vous déployez votre jeu dans plusieurs régions, gardez présent à l'esprit la manière dont les ressources peuvent interagir entre les régions. Certaines ressources, telles que les GameLift flottes Amazon, peuvent uniquement référencer d'autres ressources de la même région. Les autres

ressources, telles qu'une GameLift file d'attente Amazon, sont indépendantes de la région. Pour gérer les GameLift ressources Amazon dans plusieurs régions, nous recommandons la structure suivante.

- Packs de support régionaux : ces piles contiennent des ressources dont dépendent vos GameLift ressources Amazon. Cette pile doit inclure le compartiment S3 dans lequel vous stockez votre serveur de jeu personnalisé ou vos fichiers de script Realtime. Il peut également contenir d'autres AWS ressources pour votre jeu, telles que des tables DynamoDB, des clusters Amazon Redshift et des fonctions Lambda. La plupart de ces ressources sont spécifiques à une région, vous devez donc les créer dans chaque région. Amazon a GameLift également besoin d'un rôle IAM permettant d'accéder à ces ressources de support. Comme un rôle IAM est indépendant de la région, vous n'avez besoin que d'une seule ressource de rôle, placée dans n'importe quelle région et référencée dans toutes les autres piles de support.
- Piles Amazon GameLift régionales : cette pile contient les GameLift ressources Amazon qui doivent exister dans chaque région où votre jeu est déployé, y compris la version ou le script, un ensemble de flottes et des alias. AWS CloudFormationcrée une ressource de compilation ou de script avec des fichiers se trouvant dans un emplacement de compartiment S3 et déploie le build ou le script sur une ou plusieurs ressources du parc. Chaque flotte doit avoir un alias correspondant. La file d'attente de session de jeu fait référence à tout ou partie des alias de flotte. Vous pouvez gérer un modèle pour décrire ce type de pile et l'utiliser pour créer des ensembles de ressources identiques dans chaque région.
- GameLiftPile Amazon globale — Cette pile contient votre file d'attente de sessions de jeu et vos ressources de matchmaking. Ces ressources peuvent être situées dans n'importe quelle région et sont généralement placées dans la même région. La file d'attente peut faire référence à des flottes ou à des alias situés dans n'importe quelle région. Pour placer des files d'attente supplémentaires dans différentes régions, créez des piles globales supplémentaires.

Les diagrammes ci-dessous illustrent une structure à plusieurs piles pour le déploiement de ressources dans plusieurs régions AWS. Le premier diagramme montre une structure pour une file d'attente de session de jeu unique. Le deuxième diagramme montre une structure avec plusieurs files d'attente.


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



Mettre à jour les versions

Les GameLift versions d'Amazon sont immuables, tout comme la relation entre une version et une flotte. Par conséquent, lorsque vous mettez à jour vos ressources d'hébergement pour utiliser un nouvel ensemble de fichiers de build de jeu, les étapes suivantes doivent être exécutées :

- Créez une nouvelle build en utilisant le nouvel ensemble de fichiers (remplacement).
- Créez un nouvel ensemble de flottes pour déployer la nouvelle build de jeu (remplacement).
- Redirigez les alias pour pointer vers les nouvelles flottes (mise à jour sans interruption).

Pour plus d'informations, consultez la section Mettre à jour les comportements des ressources de la pile dans le Guide de AWS CloudFormation l'utilisateur.

Déploiement automatique des mises à jour

Lors de la mise à jour d'une pile contenant des ressources de build, de flotte et d'alias associées, le comportement AWS CloudFormation par défaut consiste à effectuer automatiquement ces étapes dans l'ordre. Vous déclenchez cette mise à jour en téléchargeant d'abord les nouveaux fichiers de build vers un nouvel emplacement S3. Ensuite, vous modifiez votre modèle de build AWS CloudFormation pour pointer vers le nouvel emplacement S3. Lorsque vous mettez à jour votre pile avec le nouvel emplacement S3, la séquence AWS CloudFormation suivante se déclenche :

1. Récupère les nouveaux fichiers depuis S3, valide les fichiers et crée une nouvelle version AmazonGameLift.
2. Mise à jour de la référence de build dans le modèle de flotte, ce qui déclenche la création d'une nouvelle flotte.
3. Lorsque les nouvelles flottes sont actives, mise à jour de la référence de flotte dans l'alias, ce qui déclenche la mise à jour de l'alias pour cibler les nouvelles flottes.
4. Suppression de l'ancienne flotte.
5. Suppression de l'ancienne build.

Si votre file d'attente de session de jeu utilise des alias de flotte, le trafic des joueurs est automatiquement basculé vers les nouvelles flottes dès que les alias sont mis à jour. Les anciennes flottes sont progressivement vidées de leurs joueurs à mesure que les sessions de jeu se terminent. La mise à l'échelle automatique gère la tâche d'ajout et de suppression des instances à partir de chaque ensemble de flottes lorsque le trafic des joueurs fluctue. Vous pouvez également spécifier un nombre d'instances initial souhaité pour accélérer rapidement la montée en charge du commutateur et activer la mise à l'échelle automatique ultérieurement.

Vous pouvez également demander que AWS CloudFormation conserve les ressources au lieu de les supprimer. Pour plus d'informations, consultez RetainResources dans la Référence d'API AWS CloudFormation.

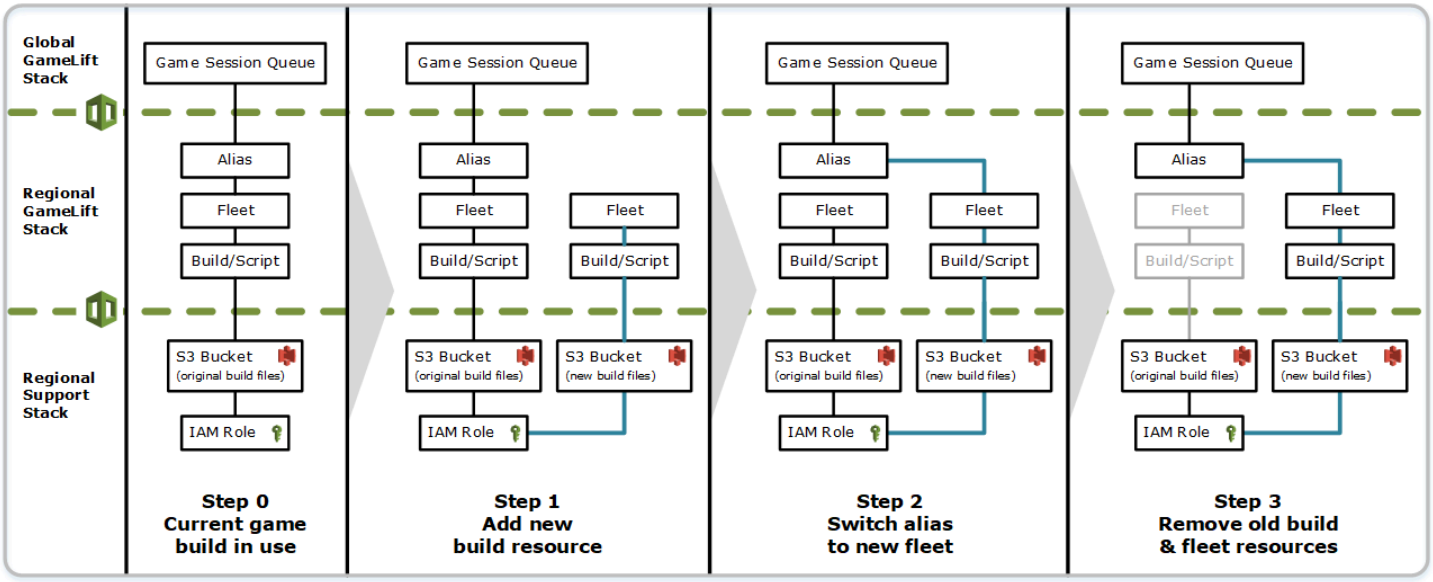
Déploiement manuel des mises à jour

Si vous souhaitez contrôler davantage le moment où les nouvelles flottes sont en ligne pour les joueurs, vous disposez de plusieurs options. Vous pouvez choisir de gérer les alias manuellement à l'aide de la GameLift console Amazon ou de l'interface de ligne de commande. Au lieu de mettre à jour votre modèle de build pour remplacer la build et les flottes, vous pouvez ajouter un deuxième ensemble de définitions de build et de flotte à votre modèle. Lorsque vous mettez à jour le modèle,

AWS CloudFormation crée une deuxième ressource de build et les flottes correspondantes. Les ressources existantes n'étant pas remplacées, elles ne sont pas supprimées et les alias continuent de pointer vers les flottes d'origine.

Le principal avantage de cette approche est qu'elle vous donne de la flexibilité. Vous pouvez créer des ressources distinctes pour la nouvelle version de votre build, tester les nouvelles ressources et contrôler le moment où les nouvelles flottes sont mises en ligne pour les joueurs. Un inconvénient potentiel est que cela nécessite deux fois plus de ressources dans chaque région pendant une brève période.

Le schéma suivant illustre ce processus.



Comment fonctionnent les annulations

Lors de l'exécution d'une mise à jour des ressources, si une étape ne se termine pas correctement, AWS CloudFormation lance automatiquement une restauration. Ce processus inverse chaque étape dans l'ordre, en supprimant les ressources nouvellement créées.

Si vous devez déclencher manuellement une restauration, restaurez l'emplacement d'origine de la clé d'emplacement S3 du modèle de build, puis mettez à jour votre pile. Une nouvelle GameLift version et une nouvelle flotte Amazon sont créées, et l'alias passe à la nouvelle flotte une fois que celle-ci est active. Si vous gérez des alias séparément, vous devez les modifier de sorte qu'ils pointent vers les nouvelles flottes.

Pour plus d'informations sur la façon de gérer une restauration qui échoue ou reste bloquée, voir Continuer à annuler une mise à jour dans le Guide de l'AWS CloudFormationutilisateur.

Peering VPC pour Amazon GameLift

Cette rubrique explique comment configurer une connexion d'appairage VPC entre vos serveurs de jeu GameLift hébergés sur Amazon et vos autres ressources extérieures à Amazon. GameLift Utilisez les connexions d'appairage Amazon Virtual Private Cloud (VPC) pour permettre à vos serveurs de jeu de communiquer directement et en privé avec vos autres AWS ressources, telles qu'un service Web ou un référentiel. Vous pouvez établir un peering VPC avec toutes les ressources qui s'exécutent sur un AWS compte auquel vous avez accès AWS et qui sont gérées par celui-ci.

> **Note**
>
> L'appairage de VPC est une fonction avancée. Pour en savoir plus sur les options préférées permettant à vos serveurs de jeu de communiquer directement et en privé avec vos autres AWS ressources, consultezCommuniquez avec les autres AWS ressources de vos flottes.

Si vous connaissez déjà Amazon VPC et le peering VPC, sachez que la configuration du peering avec les serveurs de GameLift jeu Amazon est quelque peu différente. Vous n'avez pas accès au VPC qui contient vos serveurs de jeu (il est contrôlé par le GameLift service Amazon). Vous ne pouvez donc pas directement demander le peering VPC pour ce dernier. Au lieu de cela, vous devez d'abord préautoriser le VPC avec vos GameLift ressources extérieures à Amazon pour qu'il accepte une demande de peering émanant du service Amazon. GameLift Vous demandez ensuite GameLift à Amazon de demander le peering VPC que vous venez d'autoriser. Amazon se GameLift charge des tâches de création de la connexion d'appairage, de configuration des tables de routage et de configuration de la connexion.

Pour configurer l'appairage de VPC pour une flotte existante

1. Obtenez des identifiants de AWS compte et des informations d'identification.

   Vous avez besoin d'un identifiant et d'identifiants de connexion pour les AWS comptes suivants. Vous pouvez trouver les identifiants de AWS compte en vous connectant AWS Management Consoleet en consultant les paramètres de votre compte. Pour obtenir les informations d'identification, accédez à la console IAM.

   - AWScompte que vous utilisez pour gérer vos serveurs de GameLift jeux Amazon.
   - AWScompte que vous utilisez pour gérer vos GameLift ressources autres qu'Amazon.

Si vous utilisez le même compte pour des GameLift ressources Amazon GameLift et non Amazon, vous n'avez besoin que d'un identifiant et d'informations d'identification pour ce compte.

2. Obtenez les identifiants de chaque VPC.

   Obtenez les informations suivantes pour les deux VPC à appairer :

   - VPC pour vos serveurs de GameLift jeux Amazon : il s'agit de votre identifiant de GameLift flotte Amazon. Vos serveurs de jeu sont déployés GameLift sur Amazon sur un parc d'instances EC2. Une flotte est automatiquement placée dans son propre VPC, qui est géré par le GameLift service Amazon. Vous n'avez pas d'accès direct au VPC, il est donc identifié par l'ID du parc.
   - VPC pour vos GameLift AWS ressources autres qu'Amazon : vous pouvez établir un peering VPC avec toutes les ressources qui s'exécutent sur un AWS compte auquel vous avez accès AWS et qui sont gérées par celui-ci. Si vous n'avez pas encore créé de VPC pour ces ressources, consultez Premiers pas avec Amazon VPC. Une fois que vous avez créé un VPC, vous pouvez trouver l'ID du VPC en vous connectant à AWS Management Console pour Amazon VPC et en affichant vos VPC.

   > **Note**
   >
   > Lorsque vous configurez un appairage, les deux VPC doivent exister dans la même région. Le VPC des serveurs de jeu de votre GameLift flotte Amazon se trouve dans la même région que la flotte.

3. Autorisez un appairage de VPC.

   Au cours de cette étape, vous préautorisez une future demande d'Amazon visant à associer le VPC GameLift à vos serveurs de jeu à votre VPC pour les ressources n'appartenant pas à Amazon. GameLift Cette étape met à jour le groupe de sécurité pour votre VPC.

   Pour autoriser l'appairage VPC, appelez l'API du GameLift service Amazon CreateVpcPeeringAuthorization() ou utilisez la AWS commande CLI. `create-vpc-peering-authorization` Passez cet appel en utilisant le compte qui gère vos GameLift ressources autres qu'Amazon. Identifiez les informations suivantes :

   - ID du VPC homologue : il s'agit du VPC avec vos ressources autres GameLift qu'Amazon.

- ID de GameLift AWS compte Amazon : il s'agit du compte que vous utilisez pour gérer votre GameLift flotte Amazon.

Une fois que vous avez autorisé un peering VPC, l'autorisation reste valide pendant 24 heures à moins qu'elle ne soit révoquée. Vous pouvez gérer vos autorisations d'appairage de VPC à l'aide des opérations suivantes :

- DescribeVpcPeeringAuthorizations() (AWSCLI`describe-vpc-peering-authorizations`).
- DeleteVpcPeeringAuthorization() (AWSCLI`delete-vpc-peering-authorization`).

4. Demandez une connexion d'appairage.

   Avec une autorisation valide, vous pouvez demander à Amazon d'GameLiftétablir une connexion de peering.

   Pour demander un peering VPC, appelez l'API du GameLift service Amazon CreateVpcPeeringConnection() ou utilisez la AWS commande CLI. `create-vpc-peering-connection` Passez cet appel en utilisant le compte qui gère vos serveurs de GameLift jeux Amazon. Utilisez les informations suivantes pour identifier les deux VPC que vous souhaitez appairer :

   - ID du VPC homologue et ID de AWS compte : il s'agit du VPC pour vos GameLift ressources autres qu'Amazon et du compte que vous utilisez pour les gérer. L'ID du VPC doit correspondre à celui d'une autorisation d'appairage valide.
   - Fleet ID : il permet d'identifier le VPC pour vos serveurs de GameLift jeux Amazon.

5. Suivez l'état de la connexion de l'appairage.

   La demande d'une connexion d'appairage de VPC est une opération asynchrone. Pour suivre l'état d'une demande d'appairage et gérer les cas de réussite ou d'échec, utilisez l'une des options suivantes :

   - Interrogez de façon continue avec `DescribeVpcPeeringConnections()`. Cette opération récupère l'enregistrement de connexion de l'appairage de VPC, y compris l'état de la demande. Si une connexion d'appairage est créée avec succès, l'enregistrement de la connexion contient également le bloc CIDR d'adresses IP privées attribué au VPC.
   - Gérez les événements de parc associés aux connexions d'appairage des VPC avec DescribeFleetEvents(), y compris les événements de réussite et d'échec.

Une fois la connexion d'appairage établie, vous pouvez la gérer à l'aide des opérations suivantes :

- DescribeVpcPeeringConnections() (AWSCLI`describe-vpc-peering-connections`).
- DeleteVpcPeeringConnection() (AWSCLI`delete-vpc-peering-connection`).

Pour configurer l'appairage de VPC avec une nouvelle flotte

Vous pouvez créer une nouvelle GameLift flotte Amazon et demander une connexion d'appairage VPC en même temps.

1. Obtenez des identifiants de AWS compte et des informations d'identification.

   Vous avez besoin d'un identifiant et d'identifiants de connexion pour les deux AWS comptes suivants. Vous pouvez trouver les identifiants de AWS compte en vous connectant AWS Management Consoleet en consultant les paramètres de votre compte. Pour obtenir les informations d'identification, accédez à la console IAM.

   - AWScompte que vous utilisez pour gérer vos serveurs de GameLift jeux Amazon.
   - AWScompte que vous utilisez pour gérer vos GameLift ressources autres qu'Amazon.

   Si vous utilisez le même compte pour des GameLift ressources Amazon GameLift et non Amazon, vous n'avez besoin que d'un identifiant et d'informations d'identification pour ce compte.
2. Obtenez l'ID VPC pour vos ressources autres qu'Amazon GameLiftAWS.

   Si vous n'avez pas encore créé de VPC pour ces ressources, faites-le dès maintenant (voir Premiers pas avec Amazon VPC). Veillez à créer le nouveau VPC dans la région où vous prévoyez de créer votre nouvelle flotte. Si vos GameLift ressources autres qu'Amazon sont gérées sous un AWS compte ou un utilisateur/groupe d'utilisateurs différent de celui que vous utilisez avec AmazonGameLift, vous devrez utiliser ces informations de compte lors de la demande d'autorisation à l'étape suivante.

   Une fois que vous avez créé un VPC, vous pouvez localiser l'ID du VPC dans la console Amazon VPC en affichant vos VPC.
3. Autorisez un peering VPC avec des ressources n'appartenant pas GameLift à Amazon.

   Lorsqu'Amazon GameLift crée la nouvelle flotte et le VPC correspondant, il envoie également une demande d'appariement avec le VPC pour vos ressources autres GameLift qu'Amazon.

Vous devez pré-autoriser cette demande. Cette étape met à jour le groupe de sécurité pour votre VPC.

À l'aide des informations d'identification du compte qui gèrent vos GameLift ressources autres qu'Amazon, appelez l'API du GameLift service Amazon CreateVpcPeeringAuthorization() ou utilisez la AWS commande CLI. `create-vpc-peering-authorization` Identifiez les informations suivantes :

- ID du VPC homologue : ID du VPC avec vos ressources autres GameLift qu'Amazon.
- ID de GameLift AWS compte Amazon : ID du compte que vous utilisez pour gérer votre GameLift flotte Amazon.

Une fois que vous avez autorisé un peering VPC, l'autorisation reste valide pendant 24 heures à moins qu'elle ne soit révoquée. Vous pouvez gérer vos autorisations d'appairage de VPC à l'aide des opérations suivantes :

- DescribeVpcPeeringAuthorizations() (AWSCLI`describe-vpc-peering-authorizations`).
- DeleteVpcPeeringAuthorization() (AWSCLI`delete-vpc-peering-authorization`).

4. Suivez les instructions pour créer une nouvelle flotte à l'aide de l'AWSinterface de ligne de commande. Incluez les paramètres requis supplémentaires suivants :

   - peer-vpc-aws-account-id : ID du compte que vous utilisez pour gérer le VPC avec vos ressources autres GameLift qu'Amazon.
   - peer-vpc-id— ID du VPC associé à votre GameLift compte externe.

Un appel réussi à create-fleet avec les paramètres d'appairage de VPC génère à la fois une nouvelle flotte et une nouvelle demande d'appairage de VPC. Le statut de la flotte est défini sur New et le processus d'activation de la flotte est initié. Le statut de la demande d'appairage est défini sur initiating-request. Vous pouvez suivre le succès ou l'échec de la demande de peering en appelant describe-vpc-peering-connections.

Lorsque vous demandez une connexion d'appairage de VPC avec une nouvelle flotte, les deux actions aboutissent ou échouent. Si une flotte échoue pendant le processus de création, la connexion d'appairage de VPC n'est pas établie. De même, si une connexion d'appairage de VPC échoue pour une raison quelconque, la nouvelle flotte ne peut pas passer du statut Activating au statut Active.

> **Note**
>
> La nouvelle connexion d'appairage de VPC n'est pas terminée tant que la flotte n'est pas prête à être active. Cela signifie que la connexion n'est pas disponible et ne peut pas être utilisée pendant le processus d'installation du serveur de jeu.

L'exemple suivant crée à la fois une nouvelle flotte et une connexion d'appairage entre un VPC préétabli et le VPC de la nouvelle flotte. Le VPC préétabli est identifié de manière unique par la combinaison de votre identifiant de GameLift AWS compte autre qu'Amazon et de l'ID du 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"
```

Version copiable :

```
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"
```

Résolution des problèmes d'appairage de VPC

Si vous ne parvenez pas à établir une connexion de peering VPC pour vos serveurs de GameLift jeu Amazon, considérez les causes profondes les plus courantes suivantes :

- Une autorisation pour la connexion demandée est introuvable.
  - Vérifiez l'état d'une autorisation VPC pour le VPC autre qu'AmazonGameLift. Il se peut qu'il n'existe pas ou qu'il ait expiré.
  - Vérifiez les régions des deux VPC que vous essayez d'appairer. S'ils ne sont pas dans la même région, ils ne peuvent pas être appairés.
- Les blocs d'adresse CIDR (voir Configurations de connexion d'appairage VPC non valides) de vos deux VPC se chevauchent. Les blocs CIDR IPv4 affectés aux VPC appairés ne peuvent pas se chevaucher. Le bloc d'adresse CIDR du VPC de votre GameLift flotte Amazon est automatiquement attribué et ne peut pas être modifié. Vous devez donc modifier le bloc d'adresse CIDR du VPC pour vos ressources autres qu'Amazon. GameLift Pour résoudre ce problème :
  - Recherchez ce bloc CIDR pour votre GameLift flotte Amazon en appelant`DescribeVpcPeeringConnections()`.
  - Accédez à la console Amazon VPC, recherchez le VPC correspondant à vos GameLift ressources autres qu'Amazon et modifiez le bloc d'adresse CIDR afin qu'ils ne se chevauchent pas.
- La nouvelle flotte ne s'est pas activée (lors de la demande d'appairage VPC avec une nouvelle flotte). Si la nouvelle flotte n'a pas réussi à passer au statut Actif, il n'y a pas de VPC à appairer, de sorte que la connexion d'appairage ne peut pas réussir.

Affichage de vos données de jeu sur la console

Le GameLift service Amazon géré collecte en permanence des données relatives aux jeux actifs afin de vous aider à comprendre le comportement et les performances des joueurs. Avec la GameLift console Amazon, vous pouvez consulter, gérer et analyser ces informations pour vos versions, vos flottes, vos sessions de jeu et vos sessions de joueurs.

Rubriques

Afficher votre GameLift statut Amazon actuel

Le GameLift tableau de bord Amazon fournit un aperçu des éléments suivants :

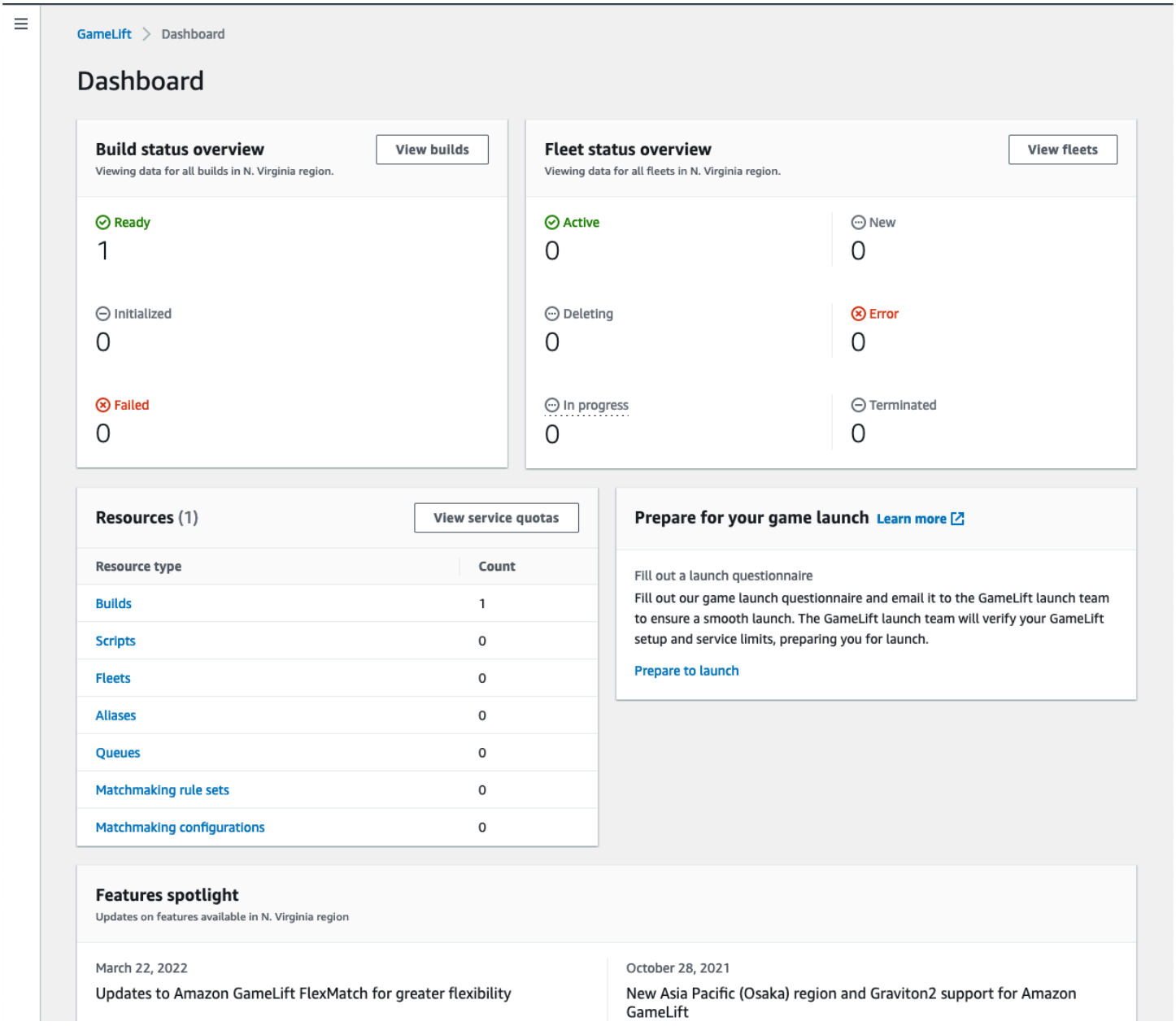
- Le nombre de builds aux statuts Prêt, Initialisé et Échoué. Choisissez Afficher les versions pour plus de détails sur les versions de votre région actuelle.
- Le nombre de flottes dans tous les statuts. Choisissez Afficher les flottes pour obtenir des informations sur les flottes de votre région actuelle.
- Vos ressources actuelles.
- Nouvelles annonces de fonctionnalités et de services.

Pour ouvrir le tableau de GameLift bord Amazon

- Dans la GameLift console Amazon, dans le volet de navigation, choisissez Dashboard.

Depuis le tableau de bord, vous pouvez :

- Préparez votre jeu pour le lancement en choisissant Préparer le lancement et en remplissant le questionnaire de lancement correspondant.
- Demandez des augmentations de quotas de service en vue des lancements ou en réponse aux lancements en choisissant Afficher les quotas de service.
- Consultez les articles de blog et les informations détaillées sur les nouvelles fonctionnalités en cliquant sur le lien dans la section Fonctionnalités.



Afficher vos constructions

Sur la page Builds de la GameLiftconsole Amazon, vous pouvez consulter des informations sur toutes les versions de serveurs de jeu que vous avez mises en ligne sur Amazon et les gérerGameLift. Dans le volet de navigation, choisissez Hosting, Builds.

La page Builds affiche les informations suivantes pour chaque build :

> **Note**
>
> La page Builds affiche uniquement les builds de votre AWS région actuelle.

- Nom : nom associé à la version téléchargée.
- État : statut de la version. Affiche l'un des trois messages d'état suivants :
  - Initialisé : le chargement n'a pas commencé ou est toujours en cours.
  - Prêt — La version est prête pour la création de la flotte.
  - Échec : le délai de compilation a expiré avant qu'Amazon ne GameLift reçoive les fichiers binaires.
- Heure de création : date et heure auxquelles vous avez chargé la version sur AmazonGameLift.
- ID de build : ID unique attribué au build lors du téléchargement.
- Version : étiquette de version associée à la version téléchargée.
- Système d'exploitation : système d'exploitation sur lequel s'exécute la version. Le système d'exploitation de compilation détermine le système d'exploitation qu'Amazon GameLift installe sur les instances d'un parc.
- Taille : taille, en mégaoctets (Mo), du fichier de compilation chargé sur Amazon. GameLift
- Flottes : nombre de flottes déployées lors de la construction.

À partir de cette page, vous pouvez effectuer l'une des opérations suivantes :

- Consulter les détails d'une version de génération. Choisissez le nom d'un build pour ouvrir la page de détails du build correspondant.
- Créer une nouvelle flotte à partir d'une version de génération. Sélectionnez une construction, puis choisissez Créer une flotte.

- Filtrer et trier la liste de versions de génération. Utiliser les commandes situées dans la partie supérieure tableau.
- Supprimer une build. Sélectionnez une version, puis choisissez Supprimer.

Détails de la construction

Sur la page Builds, choisissez le nom d'un build pour ouvrir sa page de détails. La section Vue d'ensemble de la page de détails affiche les mêmes informations de synthèse des versions que la page des versions. La section Flottes affiche la liste des flottes créées avec la version, y compris les mêmes informations récapitulatives que celles de la page Flottes.

Afficher vos scripts

Sur la page Scripts de la GameLiftconsole Amazon, vous pouvez consulter les informations relatives à tous les scripts Realtime Servers que vous avez chargés sur Amazon GameLift et les gérer. Dans le volet de navigation, choisissez Hosting, Scripts.

La page Scripts affiche les informations suivantes pour chaque script :

> **Note**
>
> La page Scripts affiche uniquement les scripts de votre AWS région actuelle.

- Nom : nom associé au script chargé.
- ID : ID unique attribué au script lors du téléchargement.
- Version : étiquette de version associée au script chargé.
- Taille : taille, en mégaoctets (Mo), du fichier script chargé sur Amazon. GameLift
- Heure de création : date et heure auxquelles vous avez chargé le script sur AmazonGameLift.
- Flottes : nombre de flottes déployées avec le script.

À partir de cette page, vous pouvez effectuer l'une des opérations suivantes :

- Afficher les détails du script. Choisissez le nom d'un build pour ouvrir la page de détails de son script.
- Créez une nouvelle flotte à partir d'un script. Sélectionnez un script, puis choisissez Create fleet.

- Filtrez et triez la liste des scripts. Utiliser les commandes situées dans la partie supérieure tableau.
- Supprimez un script. Sélectionnez un script, puis choisissez Supprimer.

Détails du script

Sur la page Scripts, choisissez le nom d'un script pour ouvrir sa page de détails. La section Vue d'ensemble de la page de détails affiche les mêmes informations récapitulatives du script que la page Builds. La section Flottes affiche la liste des flottes créées à l'aide du script, y compris les mêmes informations récapitulatives que celles de la page Flottes.

Consultez vos flottes

Vous pouvez consulter les informations relatives à toutes les flottes créées pour héberger vos jeux sur Amazon GameLift depuis votre AWS compte. La liste répertorie les flottes créées dans votre région actuelle. Sur la page Flottes, vous pouvez créer une nouvelle flotte ou consulter des informations supplémentaires sur une flotte. La page détaillée d'une flotte contient des informations d'utilisation, des statistiques, des données de session de jeu et des données de session de joueur. Vous pouvez également modifier un enregistrement de flotte ou supprimer une flotte.

Pour afficher la page Flottes, choisissez Flottes dans le volet de navigation.

La page Flottes affiche les informations récapitulatives suivantes par défaut : Vous pouvez personnaliser les informations affichées en cliquant sur le bouton Paramètres (engrenage).

- Nom — Nom convivial donné à la flotte.
- État : état de la flotte, qui peut être l'un des états suivants : Nouveau, En cours de développement, En cours de construction et Actif.
- Heure de création : date et heure de création de la flotte.
- Type de calcul : type de calcul utilisé pour héberger vos jeux. Un parc peut être un parc EC2 géré ou un parc Anywhere.
- Type d'instance : type d'instance Amazon EC2, qui détermine la capacité informatique des instances du parc.
- Instances actives : nombre d'instances EC2 utilisées pour le parc.
- Instances souhaitées : nombre d'instances EC2 à maintenir actives.
- Sessions de jeu : nombre de sessions de jeu actives dans la flotte. Les données sont retardées de cinq minutes.

Afficher les détails de la flotte

Accédez à la page détaillée d'une flotte à partir du tableau de bord ou de la page Flottes en choisissant le nom de la flotte.

Sur la page des détails de la flotte, vous pouvez effectuer les actions suivantes :

- Mettez à jour les attributs, les paramètres des ports et la configuration d'exécution d'une flotte.
- Ajoutez ou supprimez des emplacements de flotte.
- Modifier les paramètres de capacité de la flotte.
- Définissez ou modifiez la mise à l'échelle automatique du suivi des cibles.
- Supprimer une flotte.

Détails

Paramètres de la flotte

- Numéro de flotte : identifiant unique attribué à la flotte.
- Nom : nom de la flotte.
- ARN : identifiant attribué à cette flotte. L'ARN d'une flotte l'identifie comme une GameLift ressource Amazon et spécifie la région et le AWS compte.
- Description — Brève description identifiable de la flotte.
- État — État actuel de la flotte, qui peut être nouvelle, en cours de téléchargement, en construction et active.
- Heure de création : date et heure de création de la flotte.
- Heure de fin : date et heure de fin de service de la flotte. Ce champ est vide si la flotte est toujours active.
- Type de parc : indique si le parc utilise des instances à la demande ou ponctuelles.
- Type EC2 : type d'instance Amazon EC2 sélectionné pour le parc lors de sa création.
- Rôle d'instance : rôle AWS IAM qui gère l'accès à vos autres AWS ressources, si une telle ressource a été fournie lors de la création de la flotte.
- Certificat TLS : indique si la flotte est activée ou désactivée pour utiliser un certificat TLS pour authentifier un serveur de jeu et crypter toutes les communications client-serveur.
- Groupe de mesures : groupe utilisé pour agréger les mesures de plusieurs flottes.

- Politique de protection relative à l'évolutivité du jeu : Configuration actuelle de la protection des sessions de jeu pour la flotte.
- Nombre maximum de sessions de jeu par joueur : le nombre maximum de sessions qu'un joueur peut créer pendant la période de politique.
- Période de politique : combien de temps faut-il attendre avant de réinitialiser le nombre de sessions créées par un joueur.

Détails de la construction

La section Détails de la construction affiche la version hébergée sur la flotte. Sélectionnez le nom du build pour voir la page détaillée complète du build.

Configuration du moteur d'exécution

La section Configuration du runtime affiche les processus du serveur à lancer sur chaque instance. Il inclut le chemin d'accès à l'exécutable du serveur de jeux et des paramètres de lancement en option.

Activation de session de jeu

La section Activation de la session de jeu affiche le nombre de processus du serveur qui se lancent en même temps et le temps qu'il faut attendre pour que le processus s'active avant de le terminer.

Paramètres du port EC2

La section Ports affiche les autorisations de connexion de la flotte, y compris l'adresse IP et les plages de paramètres de port.

Métriques

L'onglet Métriques affiche une représentation graphique des métriques d'une flotte dans le temps. Pour plus d'informations sur l'utilisation des métriques dans AmazonGameLift, consultezSurveillez Amazon GameLift avec Amazon CloudWatch.

Événements

L'onglet Events comporte un journal de tous les événements qui se sont produits sur la flotte, avec le code d'événement, le message et l'horodatage. Consultez les descriptions des événements dans la référence des GameLift API Amazon.

Mise à l'échelle

L'onglet Scaling contient des informations sur la capacité du parc, notamment l'état actuel et l'évolution de la capacité au fil du temps. Il propose aussi des outils pour la mise à jour des limites de capacité et la gestion de la scalabilité automatique.

Capacité d'évolutivité

Consultez les paramètres de capacité actuels du parc pour chaque emplacement du parc. Pour plus d'informations sur la modification des limites et de la capacité, consultezÉlargir la capacité GameLift d'hébergement d'Amazon.

- AWSEmplacement : nom de l'emplacement où les instances de flotte sont déployées.
- État : statut d'hébergement de l'emplacement de la flotte. Le statut de l'emplacement doit `ACTIVE` être le même pour pouvoir héberger des matchs.
- Taille minimale : le plus petit nombre d'instances devant être déployées sur l'emplacement.
- Instances souhaitées : nombre cible d'instances actives pour conserver l'emplacement. Lorsque les instances actives et les instances souhaitées ne sont pas identiques, un événement de dimensionnement est lancé pour démarrer ou arrêter les instances selon les besoins jusqu'à ce que les instances actives soient égales aux instances souhaitées.
- Taille maximale : le plus grand nombre d'instances pouvant être déployées sur le site.
- Disponible : limite de service sur les instances moins le nombre d'instances utilisées. Cette valeur indique le nombre maximum d'instances que vous pouvez ajouter à l'emplacement.

Règles de mise à l'échelle automatique

Cette section contient des informations sur les politiques de mise à l'échelle automatique appliquées au parc. Vous pouvez configurer ou mettre à jour une politique basée sur des objectifs. Les politiques basées sur des règles de la flotte, qui doivent être définies à l'aide du AWS SDK ou de la CLI, sont affichées ici. Pour plus d'informations sur la mise à l'échelle, reportez-vous à la sectionAdaptez automatiquement la capacité de votre flotte avec Amazon GameLift.

Historique du dimensionnement

Affichez des graphiques de l'évolution de la capacité au fil du temps.

Emplacements

L'onglet Emplacements répertorie tous les emplacements où les instances de flotte sont déployées. Les emplacements incluent la région d'origine de la flotte et tous les sites distants qui ont été ajoutés. Vous pouvez ajouter ou supprimer des lieux directement dans cet onglet.

- Emplacement : nom de l'emplacement où les instances de flotte sont déployées.
- État : statut d'hébergement de l'emplacement de la flotte. L'état de l'emplacement suit le processus d'activation des premières instances de l'emplacement. Le statut de l'emplacement doit `ACTIVE` être le même pour pouvoir héberger des matchs.
- Instances actives : nombre d'instances dont les processus serveur s'exécutent sur l'emplacement du parc.
- Serveurs actifs : nombre de processus de serveurs de jeu capables d'héberger des sessions de jeu dans l'emplacement de la flotte.
- Sessions de jeu : nombre de sessions de jeu actives sur les instances situées dans l'emplacement de la flotte.
- Sessions de joueurs : nombre de sessions de joueurs, qui représentent des joueurs individuels, participant à des sessions de jeu actives dans l'emplacement de la flotte.

Sessions de jeu

L'onglet Game sessions affiche la liste des sessions de jeu passées et présentes hébergées dans la flotte, y compris certaines informations détaillées. Choisissez un identifiant de session de jeu pour accéder à des informations supplémentaires sur les sessions de jeu, y compris les sessions des joueurs. Pour plus d'informations sur les sessions des joueurs, consultezAfficher les données sur les sessions de jeu et les sessions des joueurs.

Afficher les données sur les sessions de jeu et les sessions des joueurs

Vous pouvez consulter des informations sur les sessions de jeu et les joueurs individuels. Pour plus d'informations sur les sessions de jeu et les sessions de joueur, consultez Comment les joueurs se connectent aux jeux.

Pour consulter la session de jeu et les données des joueurs

1. Dans la GameLiftconsole Amazon, dans le volet de navigation, choisissez Fleets.
2. Choisissez la flotte dans la liste des flottes qui a hébergé vos sessions de jeu.
3. Choisissez l'onglet Sessions de jeu. Cet onglet répertorie toutes les sessions de jeu hébergées sur la flotte ainsi que des informations récapitulatives.
4. Choisissez une session de jeu pour afficher des informations supplémentaires sur la session de jeu et la liste des joueurs connectés au jeu.

Détails

Présentation

Cette section affiche un résumé des informations de votre session de jeu.

- État — État de la session de jeu.
  - Activation — L'instance lance une session de jeu.
  - Active : une session de jeu est en cours et est disponible pour recevoir des joueurs, en fonction de la politique de création de joueurs de la session.
  - Terminé : la session de jeu est terminée.
- ARN : nom de ressource Amazon de la session de jeu.
- Nom : nom généré pour la session de jeu.
- Emplacement : emplacement où Amazon a GameLift hébergé la session de jeu.
- Heure de création : date et heure auxquelles Amazon GameLift a créé la session de diffusion.
- Heure de fin : date et heure de fin de la session de jeu.
- Nom DNS : nom d'hôte de la session de jeu.
- Adresse IP : adresse IP spécifiée pour la session de jeu.
- Port : numéro de port utilisé pour se connecter à la session de jeu.
- Identifiant du créateur : identifiant unique du joueur à l'origine de la session de jeu.
- Politique de création de sessions de joueurs : indique si la session de jeu accepte de nouveaux joueurs.
- Politique de protection relative à l'évolutivité du jeu : type de protection des sessions de jeu à définir sur toutes les nouvelles instances qu'Amazon GameLift lance dans la flotte.

Données de jeu

Des données bien formatées à envoyer à votre session de jeu au démarrage.

Propriétés du jeu

Propriétés des paires clé/valeur qui influent sur votre session de jeu.

Données de matchmaking

Le FlexMatch matchmaker JSON. Pour consulter et modifier le système de matchmaking, choisissez Afficher la configuration du matchmaking. Pour plus d'informations sur le FlexMatch matchmaking, voir Créer un matchmaking.

Sessions de joueur

Les données de session de joueur suivantes sont collectées pour chaque session de jeu :

- ID de session du joueur : identifiant attribué à la session du joueur.
- ID du joueur : identifiant unique du joueur. Choisissez cet identifiant pour obtenir des informations supplémentaires sur le joueur.
- État — État de la session du joueur. Les états possibles sont les suivants :
  - Réservé — La session du joueur a été réservée, mais les joueurs ne sont pas connectés.
  - Actif — La session du joueur est connectée au serveur de jeu.
  - Terminé — La session du joueur est terminée ; le joueur n'est plus connecté.
  - Expiration du délai : le joueur n'a pas réussi à se connecter.
- Heure de création : heure à laquelle le joueur s'est connecté à la session de jeu.
- Heure de fin : heure à laquelle le joueur s'est déconnecté de la session de jeu.
- Données du joueur : informations sur le joueur fournies lors de la création de la session de joueur.

Informations sur le joueur

Affichez des informations supplémentaires pour un joueur sélectionné, y compris une liste de tous les jeux auxquels un joueur est connecté sur toutes les flottes de la région actuelle. Ces informations incluent le statut, les heures de début, les heures de fin et le temps total de connexion pour chaque session de joueur. Vous pouvez choisir d'afficher les données des sessions de jeu et des flottes concernées.

Afficher vos alias

La page Alias affiche des informations sur les alias de flotte que vous avez créés dans votre région actuelle. Pour afficher la page des alias, choisissez Alias dans le volet de navigation.

Vous pouvez effectuer les opérations suivantes sur la page des alias :

- Créez un nouvel alias. Choisissez Create alias.
- Filtrez et triez le tableau des alias. Utiliser les commandes situées dans la partie supérieure tableau.
- Afficher des informations sur les alias. Choisissez un nom d'alias pour ouvrir la page détaillée de l'alias.
- Supprimer un alias. Choisissez un alias, puis choisissez Supprimer.

Détails de l'alias

La page de détails de l'alias affiche des informations sur l'alias.

À partir de cette page, vous pouvez :

- Modifiez un alias. Choisissez Edit (Modifier).
- Affichez les flottes que vous avez associées à l'alias.
- Supprimer un alias. Choisissez Delete (Supprimer).

Les informations détaillées d'alias sont les suivantes :

- ID : numéro unique utilisé pour identifier l'alias.
- Description : description de l'alias.
- ARN : nom de ressource Amazon de l'alias.
- Création : date et heure de création de l'alias.
- Dernière mise à jour : date et heure de la dernière mise à jour de l'alias.
- Type de routage : type de routage pour l'alias, qui peut être l'un des suivants :
  - Simple : achemine le trafic des joueurs vers un identifiant de flotte spécifié. Vous pouvez mettre à jour l'ID de flotte d'un alias à tout moment.

  - Terminal : renvoie un message au client. Par exemple, vous pouvez rediriger les joueurs qui utilisent un out-of-date client vers un endroit où ils peuvent obtenir une mise à niveau.
- Balises : paires clé/valeur utilisées pour identifier l'alias.

Afficher vos files d'attente

Vous pouvez afficher des informations sur toutes les files d'attente de placement de session de jeu existantes. La page des files d'attente affiche les files d'attente créées dans votre région actuelle. Depuis la page Queues, vous pouvez créer une file d'attente, supprimer les files d'attente existantes ou ouvrir une page d'informations détaillées pour une file d'attente sélectionnée. Chaque page de détails de la file d'attente contient les données de configuration et de métriques de la file d'attente. Pour plus d'informations sur les files d'attente, consultez Configuration des GameLift files d'attente Amazon pour le placement des sessions de jeu.

La page des files d'attente affiche les informations récapitulatives suivantes pour chaque file d'attente :

- Nom de la file d'attente : nom attribué à la file d'attente. Les demandes de nouvelles sessions de jeu utilisent ce nom pour spécifier une file d'attente.
- Délai d'attente : durée maximale, en secondes, pendant laquelle une demande de placement de session de jeu reste dans la file d'attente avant l'expiration du délai.
- Destinations dans la file d'attente : nombre de flottes répertoriées dans la configuration de la file d'attente. Amazon GameLift place de nouvelles sessions de jeu sur n'importe quel parc de véhicules dans la file d'attente.

Afficher les détails de la file

Vous pouvez accéder aux informations détaillées relatives à n'importe quelle file d'attente, notamment sur la configuration et les métriques de la file d'attente. Pour ouvrir la page de détails d'une file d'attente, accédez à la page Files d'attente et choisissez un nom de file d'attente.

La page des détails d'une file d'attente affiche un tableau récapitulatif et des onglets comportant des informations supplémentaires. Sur cette page, vous pouvez effectuer les opérations suivantes :

- Mettre à jour la configuration de la file d'attente, la liste des destinations et les stratégies de latence pour les joueurs. Choisissez Edit (Modifier).

- Supprimer une file d'attente. Une fois que vous avez supprimé une file d'attente, toutes les demandes de nouvelles sessions de jeu faisant référence à ce nom de file d'attente échoueront. Choisissez Delete (Supprimer).

> **Note**
>
> Pour restaurer une file d'attente supprimée, créez-en une nouvelle avec le nom de la file supprimée.

Détails

Présentation

La section Vue d'ensemble affiche le nom de ressource Amazon (ARN) de la file d'attente et le délai d'expiration. Vous pouvez utiliser l'ARN pour référencer la file d'attente dans d'autres actions ou zones d'AmazonGameLift. Le délai d'attente est la durée maximale, en secondes, pendant laquelle une demande de placement de session de jeu reste dans la file d'attente avant l'expiration du délai.

Notification d'événement

La section Notification d'événements répertorie la rubrique SNS sur laquelle Amazon GameLift publie des notifications d'événements ainsi que les données d'événement ajoutées à tous les événements créés par cette file d'attente.

Étiquettes

Le tableau des balises affiche les clés et les valeurs utilisées pour baliser la ressource. Pour plus d'informations sur le balisage, consultez la section Ressources sur le balisage. AWS

Métriques

L'onglet Metrics affiche une représentation graphique des métriques d'une file d'attente au fil du temps.

Les statistiques de file d'attente incluent une série d'informations décrivant les activités de placement dans la file d'attente, y compris les placements réussis organisés par région. Vous pouvez utiliser les données de région pour savoir où vous hébergez vos jeux. Les mesures de placement régionales peuvent aider à détecter les problèmes liés à la conception globale de la file d'attente.

Les métriques de file d'attente sont également disponibles sur AmazonCloudWatch. Pour obtenir une description des mesures disponibles, reportez-vous à la sectionGameLiftMétriques Amazon pour les files d'attente.

Destinations

L'onglet Destinations affiche toutes les flottes ou tous les alias répertoriés pour la file d'attente.

Lorsqu'Amazon GameLift recherche les destinations des ressources disponibles pour héberger une nouvelle session de jeu, il recherche l'ordre par défaut indiqué ici. Tant qu'il y a de la capacité sur la première destination répertoriée, Amazon y GameLift place de nouvelles sessions de jeu. Vous pouvez faire en sorte que les demandes de placement de session de jeu remplacent l'ordre par défaut en fournissant les données de latence du joueur. Ces données indiquent GameLift à Amazon de rechercher une destination disponible présentant la latence moyenne la plus faible pour les joueurs. Pour plus d'informations sur la conception de vos files d'attente, consultezConception d'une file d'attente de sessions de jeu.

Placement des sessions

Politiques de latence des joueurs

La section Politiques de latence du lecteur répertorie toutes les politiques utilisées par la file d'attente. Le tableau répertorie les politiques dans l'ordre dans lequel elles sont appliquées.

Emplacements

La section Emplacements indique les emplacements dans lesquels cette file d'attente peut placer une session de jeu.

Priorité

La section Priorité indique l'ordre dans lequel la file d'attente évalue les détails d'une session de jeu.

Ordre de localisation

La section Ordre de localisation indique l'ordre par défaut utilisé par la file d'attente pour placer des sessions de jeu. La file d'attente utilise cet ordre si vous n'avez pas défini d'autres types de priorité.

Surveillance d'Amazon GameLift

Si vous utilisez Amazon GameLift FleetIQ en tant que fonctionnalité autonome avec Amazon EC2, consultez la section Sécurité dans Amazon EC2 dans le guide de l'utilisateur d'Amazon EC2.

La surveillance joue un rôle important dans le maintien de la fiabilité, de la disponibilité et des performances d'Amazon GameLift et de vos autres AWS solutions. Les métriques d'Amazon sont principalement utilisées à trois fins GameLift : pour surveiller l'état du système et configurer des alarmes, pour suivre les performances et l'utilisation des serveurs de jeu, et pour gérer la capacité à l'aide d'un dimensionnement manuel ou automatique.

AWS fournit les outils de surveillance suivants pour surveiller Amazon GameLift, signaler un problème et prendre des mesures automatiques le cas échéant :

- GameLift Console Amazon
- Amazon CloudWatch —— Vous pouvez surveiller GameLift les statistiques Amazon en temps réel, ainsi que les mesures relatives aux autres AWS ressources et applications que vous exécutez sur AWS des services. CloudWatch propose une suite de fonctionnalités de surveillance, notamment des outils permettant de créer des tableaux de bord personnalisés et la possibilité de définir des alarmes qui signalent ou prennent des mesures lorsqu'une métrique atteint un seuil spécifié.
- AWS CloudTrail— capture tous les appels d'API et les événements connexes effectués par ou au nom de votre AWS compte pour Amazon GameLift et d'autres AWS services. Les données sont fournies sous forme de fichiers journaux dans un compartiment Amazon S3 que vous spécifiez. Vous pouvez identifier les utilisateurs et les comptes appelés AWS, l'adresse IP source à partir de laquelle les appels ont été effectués et la date des appels.
- Journaux de session de jeu : vous pouvez générer des messages de serveur personnalisés pour vos sessions de jeu dans des fichiers journaux stockés dans Amazon S3.

Rubriques

Surveillez Amazon GameLift avec Amazon CloudWatch

Vous pouvez surveiller Amazon GameLift à l'aide d'AmazonCloudWatch, un AWS service qui collecte des données brutes et les traite en mesures lisibles en temps quasi réel. Ces statistiques sont conservées pendant 15 mois afin de fournir une perspective historique des performances de votre serveur de jeu GameLift hébergé sur Amazon. Vous pouvez définir des alarmes qui surveillent certains seuils et envoient des notifications ou effectuent des actions spécifiques lorsque ces seuils sont atteints. Pour plus d'informations, consultez le guide de CloudWatch l'utilisateur Amazon.

Les tableaux suivants répertorient les mesures et les dimensions d'AmazonGameLift. Toutes les métriques disponibles dans le CloudWatch sont également dans la GameLift console Amazon, qui fournit les données sous la forme d'un ensemble de graphiques personnalisables. Pour accéder aux métriques CloudWatch de vos jeux, vous pouvez utiliser AWS Management Console, l'AWS CLI ou l'API CloudWatch.

Si une métrique n'a pas de position, elle utilise la position d'origine.

Dimensions pour Amazon GameLift Metrics

Amazon GameLift prend en charge le filtrage des métriques selon les dimensions suivantes.

Dimension	Description
`Location`	Filtrez les métriques pour un lieu de déploiement de flotte. Si une métrique n'a pas de position, elle utilise la position d'origine.
`FleetId`	Filtrez les métriques d'une seule flotte. Cette dimension est utilisée avec toutes les métriques de flotte relatives aux instances, aux processus serveur, aux sessions de jeu et aux sessions de joueur.
`MetricGroup`	Filtrez les métriques pour une collection de flottes. Ajoutez une flotte à un groupe de mesures en ajoutant le nom du groupe de mesures aux attributs de la flotte (voir UpdateFleetAttributes()). Cette dimension est utilisée avec toutes les métriques de

Dimension	Description
	flotte relatives aux instances, aux processus serveur, aux sessions de jeu et aux sessions de joueur.
`QueueName`	Filtrez les métriques pour une seule file d'attente . Cette dimension est utilisée avec les métriques relatives aux files d'attente de placement de session de jeu uniquement.
`ConfigurationName`	Filtrez les métriques pour une configuration de mise en relation unique. Cette dimension est utilisée avec les métriques pour les configurations de mise en relation.
`ConfigurationName-RuleName`	Filtrez les métriques pour obtenir l'intersection d'une configuration de mise en relation et d'une règle de mise en relation. Cette dimension est utilisée avec les métriques pour les règles de mise en relation uniquement.
`InstanceType`	Filtrez les métriques pour la désignation d'un type d'instance EC2, par exemple « c4.large ». Cette dimension est utilisée avec les métriques pour les instances Spot.
`OperatingSystem`	Filtrez les métriques du système d'exploitation d'une instance Cette dimension est utilisée avec les métriques des instances Spot.
`GameServerGroup`	Filtrez les statistiques de FleetIQ pour un groupe de serveurs de jeu.

GameLiftMétriques Amazon pour les flottes

L'espace de nom `AWS/GameLift` inclut les métriques suivantes associées à une activité sur un parc ou un groupe de parcs. Les flottes sont utilisées avec une GameLift solution Amazon gérée. Le GameLift service Amazon envoie des métriques à CloudWatch chaque minute.

instances

Métrique	Description
`ActiveInstances`	Instances avec l'état ACTIVE, ce qui signifie qu'elles exécutent des processus serveur actifs. Le compte inclut des instances inactives et celles qui hébergent une ou plusieurs sessions de jeu. Cette métrique mesure la capacité totale actuelle de l'instance. Cette métrique peut être utilisée avec le dimensionnement automatique. Unités : nombre CloudWatchStatistiques pertinentes : moyenne, minimale, maximale Dimensions : Emplacement
`DesiredInstances`	Nombre cible d'instances actives qu'Amazon GameLift s'efforce de maintenir dans la flotte. Grâce au dimensionnement automatique, cette valeur est déterminée en fonction des stratégie s de dimensionnement en vigueur actuellement. Sans dimensionnement automatique, cette valeur est définie manuellement. Cette métrique n'est pas disponible lorsque vous affichez les données pour les groupes de métriques du parc. Unités : nombre CloudWatchStatistiques pertinentes : moyenne, minimale, maximale Dimensions : Emplacement
`IdleInstances`	Instances actives qui hébergent actuellement zéro (0) session de jeu. Cette métrique mesure la capacité

Métrique	Description
	disponible mais non utilisée. Cette métrique peut être utilisée avec le dimensionnement automatique. Unités : nombre CloudWatchStatistiques pertinentes : moyenne, minimale, maximale Dimensions : Emplacement
`MaxInstances`	Nombre maximal d'instances autorisées pour le parc. Le nombre maximal d'instances d'un parc détermine le plafond de capacité lors du dimensionnement à la hausse manuel ou automatique. Cette métrique n'est pas disponible lorsque vous affichez les données pour les groupes de métriques du parc. Unités : nombre CloudWatchStatistiques pertinentes : moyenne, minimale, maximale Dimensions : Emplacement
`MinInstances`	Nombre minimal d'instances autorisées pour le parc. Le nombre minimal d'instances d'une flotte détermine le plancher de capacité lors de la réduction manuelle ou automatique de la capacité. Cette métrique n'est pas disponible lorsque vous affichez les données pour les groupes de métriques du parc. Unités : nombre CloudWatchStatistiques pertinentes : moyenne, minimale, maximale Dimensions : Emplacement

Métrique	Description
`PercentIdleInstances`	Pourcentage d'instances actives qui sont inactives (calculé selon la formule `IdleInstances / ActiveInstances`). Cette métrique peut être utilisée dans le cadre d'un dimensionnement automatique. Unités : pourcentage CloudWatchStatistiques pertinentes : moyenne, minimale, maximale Dimensions : Emplacement
`RecycledInstances`	Nombre d'instances spot qui ont été recyclées et remplacées. Amazon GameLift recycle les instances spot qui n'hébergent pas actuellement de sessions de jeu et qui présentent une forte probabilité d'interruption. Unités : nombre CloudWatchStatistiques pertinentes : Somme, Moyenne, Minimum, Maximum Dimensions : Emplacement
`InstanceInterruptions`	Nombre d'instances Spot qui ont été interrompues. Unités : nombre CloudWatchStatistiques pertinentes : Somme, Moyenne, Minimum, Maximum Dimensions : Emplacement

Métrique	Description
`CPUUtilization`	Métrique EC2. Pour Amazon, GameLift cette métrique représente les performances matérielles de toutes les instances actives d'un parc. Pourcentage du temps processeur physique qu'Amazon EC2 utilise pour exécuter l'instance, qui inclut le temps passé à exécuter à la fois le code utilisateur et le code Amazon EC2. Les outils de votre système d'exploitation peuvent afficher un pourcentage différent de celui CloudWatch dû à des facteurs tels que la simulation d'appareils existants, la configuration d'appareils non existants, les charges de travail nécessitant de nombreuses interruptions, la migration en direct et les mises à jour en direct. Unités : pourcentage
`NetworkIn`	Métrique EC2. Pour Amazon, GameLift cette métrique représente les performances matérielles de toutes les instances actives d'un parc. Nombre d'octets reçus par l'instance sur toutes les interfaces réseau. Cette métrique identifie le volume du trafic réseau entrant d'une application sur une seule instance. Unités : octets
`NetworkOut`	Métrique EC2. Pour Amazon, GameLift cette métrique représente les performances matérielles de toutes les instances actives d'un parc. Nombre d'octets envoyés par l'instance sur toutes les interfaces réseau. Cette métrique identifie le volume du trafic réseau sortant d'une application sur une seule instance. Unités : octets

Métrique	Description
DiskReadBytes	Métrique EC2. Pour Amazon, GameLift cette métrique représente les performances matérielles de toutes les instances actives d'un parc. Octets lus à partir de tous les volumes de stockage d'instance disponibles pour l'instance. Cette métrique permet de déterminer le volume de données que l'application lit à partir du disque dur de l'instance. Vous pouvez l'utiliser pour déterminer la vitesse de l'application. Unités : octets
DiskWriteBytes	Métrique EC2. Pour Amazon, GameLift cette métrique représente les performances matérielles de toutes les instances actives d'un parc. Octets écrits dans tous les volumes de stockage d'instanc e disponibles pour l'instance. Cette métrique permet de déterminer le volume de données que l'applica tion écrit sur le disque dur de l'instance. Vous pouvez l'utiliser pour déterminer la vitesse de l'application. Unités : octets
DiskReadOps	Métrique EC2. Pour Amazon, GameLift cette métrique représente les performances matérielles de toutes les instances actives d'un parc. Opérations de lecture terminées de tous les volumes de stockage d'instance disponibles pour l'instance, au cours de la période spécifiée. Pour calculer la moyenne d'opérations d'I/O pour la période, divisez le nombre total d'opérations de la période par le nombre de secondes de la période. Unités : nombre

Métrique	Description
DiskWriteOps	Métrique EC2. Pour Amazon, GameLift cette métrique représente les performances matérielles de toutes les instances actives d'un parc. Opérations d'écriture terminées dans tous les volumes de stockage d'instance disponibles pour l'instance, au cours de la période spécifiée. Pour calculer la moyenne d'opérations d'I/O pour la période, divisez le nombre total d'opérations de la période par le nombre de secondes de la période. Unités : nombre

Processus serveur

Métrique	Description
ActiveServerProcesses	Processus serveur avec l'état ACTIVE, ce qui signifie qu'ils fonctionnent et sont en mesure d'héberger des sessions de jeu. Le compte inclut des processus serveur inactifs et ceux qui hébergent des sessions de jeu. Cette métrique mesure la capacité totale actuelle de processus serveur. Unités : nombre CloudWatchStatistiques pertinentes : moyenne, minimale, maximale Dimensions : Emplacement
HealthyServerProcesses	Processus serveur actifs signalés comme sains. Cette métrique est utile pour le suivi de la santé générale des serveurs de jeu du parc. Unités : nombre

Métrique	Description
	CloudWatchStatistiques pertinentes : moyenne, minimale, maximale Dimensions : Emplacement
`PercentHealthyServerProcesses`	Pourcentage de l'ensemble des processus serveur actifs signalés comme sains (calculé selon la formule `HealthyServerProcesses / ActiveServerProcesses`). Unités : pourcentage CloudWatchStatistiques pertinentes : moyenne, minimale, maximale Dimensions : Emplacement
`ServerProcessAbnormalTerminations`	Processus serveur ayant été arrêtés en raison de circonstances anormales depuis le dernier rapport. Cette métrique inclut les résiliations initiées par le GameLift service Amazon. Cela se produit lorsqu'un processus serveur cesse de répondre, signale régulièrement des échecs de vérification de l'état ou ne se termine pas correctement (en appelant ProcessEnding()). Unités : nombre CloudWatchStatistiques pertinentes : Somme, Moyenne, Minimum, Maximum Dimensions : Emplacement

Métrique	Description
`ServerProcessActivations`	Processus serveur passés avec succès de l'état ACTIVATING à l'état ACTIVE depuis le dernier rapport. Les processus serveur ne peuvent pas héberger de sessions de jeu tant qu'ils ne sont pas actifs. Unités : nombre CloudWatchStatistiques pertinentes : Somme, Moyenne, Minimum, Maximum Dimensions : Emplacement
`ServerProcessTerminations`	Processus serveur arrêtés depuis le dernier rapport. Cela concerne tous les processus serveur qui, pour n'importe quelle raison, ont opéré une transition vers l'état TERMINATED, y compris en cas de mise hors service normale ou anormale du processus. Unités : nombre CloudWatchStatistiques pertinentes : Somme, Moyenne, Minimum, Maximum Dimensions : Emplacement

Sessions de jeu

Métrique	Description
`ActivatingGameSessions`	Sessions de jeu avec l'état ACTIVATING, ce qui signifie qu'elles sont en cours de démarrage. Les sessions de jeu ne peuvent pas héberger des joueurs tant qu'elles ne sont pas actives. Des chiffres élevés pendant une période de temps continue peuvent indiquer que les sessions de jeu

Métrique	Description
	ne parviennent pas à passer de l'état ACTIVATING à l'état ACTIVE. Cette métrique peut être utilisée avec le dimensionnement automatique. Unités : nombre CloudWatchStatistiques pertinentes : moyenne, minimale, maximale Dimensions : Emplacement
`ActiveGameSessions`	Sessions de jeu avec l'état ACTIVE, ce qui signifie qu'elles sont en mesure d'héberger des joueurs, et qu'elles hébergent actuellement zéro joueur ou plus. Cette métrique mesure le nombre total de sessions de jeu hébergées actuellement. Cette métrique peut être utilisée avec le dimensionnement automatique. Unités : nombre CloudWatchStatistiques pertinentes : moyenne, minimale, maximale Dimensions : Emplacement

Métrique	Description
`AvailableGameSessions`	Processus de serveur actifs et sains qui ne sont pas actuellement utilisés pour héberger une session de jeu et qui peuvent démarrer une nouvelle session de jeu sans délai pour lancer de nouveaux processus ou instances de serveur. Cette métrique peut être utilisée avec le dimensionnement automatique. **Note** Pour les flottes qui limitent les activations de sessions de jeu simultanées, utilisez la métrique. `ConcurrentActivatableGameSessions` Cette métrique représente plus précisément le nombre de nouvelles sessions de jeu qui peuvent démarrer sans aucun délai. Unités : nombre CloudWatchStatistiques pertinentes : moyenne, minimale, maximale Dimensions : Emplacement

Métrique	Description
`ConcurrentActivatableGameSessions`	Processus de serveur actifs et sains qui ne sont pas actuellement utilisés pour héberger une session de jeu et qui peuvent démarrer immédiatement une nouvelle session de jeu. Cette métrique diffère de la suivante : elle ne prend pas `AvailableGameSessions` en compte les processus du serveur qui ne peuvent actuellement pas activer une nouvelle session de jeu en raison des limites d'activations de sessions de jeu. (Voir le paramètre RuntimeConfigurationoptionnel de la flotte`MaxConcurrentGameSessionActivations`). Pour les flottes qui ne limitent pas le nombre d'activations de sessions de jeu, cette métrique est identique à. `AvailableGameSessions` Unités : nombre CloudWatchStatistiques pertinentes : moyenne, minimale, maximale Dimensions : Emplacement

Métrique	Description
`PercentAvailableGameSessions`	Pourcentage d'emplacements de session de jeu actifs sur l'ensemble des processus serveur actifs (sains ou non sains) actuellement inutilisés (calculé selon la formule `AvailableGameSessions / [ActiveGameSessions + AvailableGameSessions + unhealthy server processes]`). Cette métrique peut être utilisée avec le dimensionnement automatique. Unités : pourcentage CloudWatchStatistiques pertinentes : moyenne Dimensions : Emplacement
`GameSessionInterruptions`	Nombre de sessions de jeu ou d'instances Spot qui ont été interrompues. Unités : nombre CloudWatchStatistiques pertinentes : Somme, Moyenne, Minimum, Maximum Dimensions : Emplacement

Sessions de joueur

Métrique	Description
`CurrentPlayerSessions`	Sessions de joueur avec soit l'état ACTIVE (le joueur est connecté à une session de jeu active), soit l'état RESERVED (le joueur s'est vu attribuer un emplacement dans une session de jeu mais ne s'est pas encore connecté). Cette métrique peut être utilisée avec le dimensionnement automatique.

Métrique	Description
	Unités : nombre CloudWatchStatistiques pertinentes : moyenne, minimale, maximale
`PlayerSessionActivations`	Sessions de joueur qui sont passées de l'état RESERVED à l'état ACTIVE depuis le dernier rapport. Cette situation survient lorsqu'un joueur réussit à se connecter à une session de jeu active. Unités : nombre CloudWatchStatistiques pertinentes : Somme, Moyenne, Minimum, Maximum

GameLiftMétriques Amazon pour les files d'attente

L'espace de nom `Amazon GameLift` inclut les métriques suivantes associées à une activité sur une file d'attente de placement de session de jeu. Les files d'attente sont utilisées avec une GameLift solution Amazon gérée. Le GameLift service Amazon envoie des métriques à CloudWatch chaque minute.

Métrique	Description
`AverageWaitTime`	Temps d'attente moyen pour satisfaire les demandes de placement de session de jeu dans la file d'attente avec l'état PENDING. Unités : secondes CloudWatchStatistiques pertinentes : moyenne, minimum, maximum, somme Dimensions : Emplacement
`FirstChoiceNotViable`	Sessions de jeu correctement placées mais PAS dans la flotte de premier choix, car cette dernière

Métrique	Description
	a été considérée comme non viable (comme un parc d'instances Spot avec un taux d'interruption élevé). Cette métrique est basée sur le coût et non sur la latence. La flotte de premier choix est soit la première flotte répertoriée dans la file d'attente, soit, lorsqu'une demande de placement inclut des données de latence des joueurs, la première flotte sélectionnée par FleetIQ par ordre de priorité. S'il n'y a pas de parcs d'instances Spot viables, n'importe quel parc de cette région peut être sélectionné. Unités : nombre CloudWatchStatistiques pertinentes : moyenne, minimum, maximum, somme
`FirstChoiceOutOfCapacity`	Sessions de jeu correctement placées mais PAS dans la flotte de premier choix, car cette dernière n'avait pas de ressources disponibles. La flotte de premier choix est soit la première flotte répertori ée dans la file d'attente, soit, lorsqu'une demande de placement inclut des données de latence des joueurs, la première flotte sélectionnée selon la priorité que vous avez définie pour FleetIQ. Unités : nombre CloudWatchStatistiques pertinentes : moyenne, minimum, maximum, somme

Métrique	Description
`LowestLatencyPlacement`	Sessions de jeu correctement placées dans une région qui offre la plus faible latence possible de la file d'attente pour les joueurs. Cette métrique est émis uniquement lorsque les données de latence des joueurs sont incluses dans la requête de placement. Unités : nombre CloudWatchStatistiques pertinentes : moyenne, minimum, maximum, somme
`LowestPricePlacement`	Des sessions de jeu qui ont été placées avec succès dans une flotte au prix le plus bas possible dans la file d'attente pour la région sélectionnée. Cette flotte peut être un parc d'instances Spot ou une instance à la demande si la file d'attente ne dispose d'aucune instance Spot. Cette métrique est émis uniquemen t lorsque les données de latence des joueurs sont incluses dans la requête de placement. Unités : nombre CloudWatchStatistiques pertinentes : moyenne, minimum, maximum, somme
`Placement <region name>`	Sessions de jeu correctement placées dans des flottes situés dans la région spécifiée. Cette métrique décortique la métrique `PlacementsSucceeded` par région. Unités : nombre CloudWatchStatistiques pertinentes : Somme

Métrique	Description
PlacementsCanceled	Demandes de placement de session de jeu ayant été annulées avant expiration depuis le dernier rapport. Unités : nombre CloudWatchStatistiques pertinentes : moyenne, minimum, maximum, somme
PlacementsFailed	Requêtes de placement de session de jeu ayant échoué pour une quelconque raison depuis le dernier rapport. Unités : nombre CloudWatchStatistiques pertinentes : moyenne, minimum, maximum, somme
PlacementsStarted	Nouvelle demandes de placement de session de jeu ayant été ajoutées à la file d'attente depuis le dernier rapport. Unités : nombre CloudWatchStatistiques pertinentes : moyenne, minimum, maximum, somme
PlacementsSucceeded	Demandes de placement de session de jeu ayant abouti au lancement d'une nouvelle session de jeu depuis le dernier rapport. Unités : nombre CloudWatchStatistiques pertinentes : moyenne, minimum, maximum, somme

Métrique	Description
`PlacementsTimedOut`	Demandes de placement de session de jeu ayant atteint la limite d'expiration de la file d'attente sans être exécutées depuis le dernier rapport. Unités : nombre CloudWatchStatistiques pertinentes : moyenne, minimum, maximum, somme
`QueueDepth`	Nombre de demandes de placement de session de jeu dans la file d'attente avec l'état PENDING. Unités : nombre CloudWatchStatistiques pertinentes : moyenne, minimum, maximum, somme Dimensions : Emplacement

GameLiftMétriques Amazon pour le matchmaking

L'espace de `Amazon GameLift` nommage inclut des métriques sur FlexMatch l'activité pour les configurations de matchmaking et les règles de matchmaking. FlexMatchle matchmaking est utilisé avec une GameLift solution Amazon gérée. Le GameLift service Amazon envoie des métriques à CloudWatch chaque minute.

Pour plus d'informations sur la séquence des activités de matchmaking, consultez Comment GameLift FlexMatch fonctionne Amazon.

Configurations de mise en relation

Métrique	Description
`CurrentTickets`	Demandes de mise en relation en cours de traitemen t ou en attente de traitement. Unités : nombre

Métrique	Description
	CloudWatchStatistiques pertinentes : moyenne, minimum, maximum, somme
`MatchAcceptancesTimedOut`	Pour les configurations de mise en relation qui nécessitent l'acceptation, les rencontres éventuelles qui ont dépassé le délai lors de l'acceptation depuis le dernier rapport. Unités : nombre CloudWatchStatistiques pertinentes : Somme
`MatchesAccepted`	Pour les configurations de mise en relation qui nécessitent l'acceptation, les rencontres éventuelles qui ont été acceptées depuis le dernier rapport. Unités : nombre CloudWatchStatistiques pertinentes : Somme
`MatchesCreated`	Mises en relation potentielles ayant été créées depuis le dernier rapport. Unités : nombre CloudWatchStatistiques pertinentes : Somme
`MatchesPlaced`	Mises en relation placées avec succès dans une session de jeu depuis le dernier rapport. Unités : nombre CloudWatchStatistiques pertinentes : Somme

Métrique	Description
MatchesRejected	Pour les configurations de mise en relation qui nécessitent l'acceptation, les rencontres éventuelles qui ont été refusées par au moins un joueur depuis le dernier rapport. Unités : nombre CloudWatchStatistiques pertinentes : Somme
PlayersStarted	Joueurs dans des tickets de mise en relation ajoutés depuis le dernier rapport. Unités : nombre CloudWatchStatistiques pertinentes : Somme
TicketsFailed	Demandes de mise en relation ayant entraîné un échec depuis le dernier rapport. Unités : nombre CloudWatchStatistiques pertinentes : Somme
TicketsStarted	Nouvelles demandes de mise en relation ayant été créées depuis le dernier rapport. Unités : nombre CloudWatchStatistiques pertinentes : Somme
TicketsTimedOut	Demandes de mise en relation ayant dépassé le délai depuis le dernier rapport. Unités : nombre CloudWatchStatistiques pertinentes : Somme

Métrique	Description
`TimeToMatch`	Pour les demandes de mise en relation qui ont été placées dans une mise en relation potentielle avant le dernier rapport, la quantité de temps entre la création du ticket et la création de la mise en relation potentielle. Unités : secondes CloudWatchStatistiques pertinentes : échantillons de données, moyenne, minimum, maximum
`TimeToTicketCancel`	Pour les demandes de mise en relation qui ont été annulées avant le dernier rapport, la quantité de temps entre la création du ticket et l'annulation. Unités : secondes CloudWatchStatistiques pertinentes : échantillons de données, moyenne, minimum, maximum
`TimeToTicketSuccess`	Pour les demandes de mise en relation qui ont réussi avant le dernier rapport, la quantité de temps entre la création du ticket et le e placement de mise en relation réussi. Unités : secondes CloudWatchStatistiques pertinentes : échantillons de données, moyenne, minimum, maximum

Règles de mise en relation

Métrique	Description
`RuleEvaluationsPassed`	Évaluations des règles pendant le processus de mise en relation qui ont réussi depuis le dernier rapport. Cette métrique est limitée aux 50 premières règles. Unités : nombre CloudWatchStatistiques pertinentes : Somme
`RuleEvaluationsFailed`	Évaluations des règles pendant l'activité de mise en relation qui ont échoué depuis le dernier rapport. Cette métrique est limitée aux 50 premières règles. Unités : nombre CloudWatchStatistiques pertinentes : Somme

GameLiftMétriques Amazon pour FleetIQ

L'`Amazon GameLift`espace de nommage inclut des statistiques relatives au groupe de serveurs de jeu FleetIQ et à l'activité des serveurs de jeu dans le cadre d'une solution autonome FleetIQ pour l'hébergement de jeux. Le GameLift service Amazon envoie des métriques à CloudWatch chaque minute. Consultez également la section Surveillance de vos groupes et instances Auto Scaling à l'aide d'Amazon CloudWatch dans le Guide de l'utilisateur d'Amazon EC2 Auto Scaling.

Métrique	Description
`AvailableGameServers`	Serveurs de jeux qui sont disponibles pour exécuter une exécution de jeu et qui ne sont pas actuellement occupés par le jeu. Ce nombre inclut les serveurs de jeux qui ont été demandés, mais qui sont toujours à l'état AVAILABLE (DISPONIBLE). Unités : nombre

Métrique	Description
	CloudWatchStatistiques pertinentes : Somme Dimensions : GameServerGroup
`UtilizedGameServers`	Serveurs de jeux qui sont actuellement occupés avec le gameplay. Ce nombre inclut les serveurs de jeux dont l'état est UTILIZED (UTILISÉ). Unités : nombre CloudWatchStatistiques pertinentes : Somme Dimensions : GameServerGroup
`DrainingAvailableGameServers`	Serveurs de jeux sur des instances dont la résiliation est planifiée et qui ne prennent actuellement pas en charge le gameplay. En cas de nouvelle demande, la priorité de demande de ces serveurs de jeux est la plus faible. Unités : nombre CloudWatchStatistiques pertinentes : Somme Dimensions : GameServerGroup
`DrainingUtilizedGameServers`	Serveurs de jeux sur des instances dont la résiliation est planifiée et qui prennent actuellement en charge le gameplay. Unités : nombre CloudWatchStatistiques pertinentes : Somme Dimensions : GameServerGroup

Métrique	Description
`PercentUtilizedGameServers`	Partie des serveurs de jeux qui prennent actuellem ent en charge les exécutions de jeux. Cette métrique indique la capacité du serveur de jeux actuellement utilisé. Elle est utile pour mener une stratégie Auto Scaling qui peut ajouter et supprimer dynamiquement des instances afin de correspondre à la demande des joueurs. Unités : pourcentage CloudWatchStatistiques pertinentes : moyenne, minimale, maximale Dimensions : GameServerGroup
`GameServerInterruptions`	Serveurs de jeux sur des instances Spot qui ont été interrompus en raison d'une disponibilité limitée des instances Spot. Unités : nombre CloudWatchStatistiques pertinentes : Somme Dimensioni :GameServerGroup, InstanceType
`InstanceInterruptions`	Instances Spot interrompues en raison d'une disponibilité limitée. Unités : nombre CloudWatchStatistiques pertinentes : Somme Dimensioni :GameServerGroup, InstanceType

Enregistrement des appels GameLift d'API Amazon avec AWS CloudTrail

Amazon GameLift est intégré à AWS CloudTrail un service qui fournit un enregistrement des actions effectuées par un utilisateur, un rôle ou un AWS service sur AmazonGameLift. CloudTrailcapture tous les appels d'API pour Amazon GameLift sous forme d'événements. Les appels capturés incluent les appels provenant de la GameLift console Amazon et les appels de code vers les opérations de GameLift l'API Amazon. Si vous créez un journal, vous pouvez activer la diffusion continue d'CloudTrailévénements vers un compartiment Amazon S3, y compris des événements pour AmazonGameLift. Si vous ne configurez pas de journal d'activité, vous pouvez toujours afficher les événements les plus récents dans la console CloudTrail dans Historique des événements. À l'aide des informations collectées parCloudTrail, vous pouvez déterminer la demande qui a été faite à AmazonGameLift, l'adresse IP à partir de laquelle la demande a été faite, qui l'a faite, quand elle a été faite et des détails supplémentaires.

Pour en savoir plus sur CloudTrail, veuillez consulter le Guide de l'utilisateur AWS CloudTrail.

GameLiftInformations sur Amazon dans CloudTrail

CloudTrail est activé sur votre Compte AWS lorsque vous créez le compte. Lorsqu'une activité se produit sur AmazonGameLift, cette activité est enregistrée dans un CloudTrail événement avec d'autres événements de AWS service dans l'historique des événements. Vous pouvez afficher, rechercher et télécharger les événements récents dans votre Compte AWS. Pour plus d'informations, consultez Affichage des événements avec l'historique des événements CloudTrail.

Pour un enregistrement continu des événements enregistrés dans votre compteCompte AWS, y compris des événements pour AmazonGameLift, créez un historique. Un journal permet CloudTrail de transmettre des fichiers journaux à un compartiment Amazon S3. Par défaut, lorsque vous créez un journal d'activité dans la console, il s'applique à toutes les régions Régions AWS. Le journal d'activité consigne les événements de toutes les Régions dans la partition AWS et livre les fichiers journaux dans le compartiment Simple Storage Service (Amazon S3) de votre choix. En outre, vous pouvez configurer d'autres services AWS pour analyser plus en profondeur les données d'événement collectées dans les journaux CloudTrail et agir sur celles-ci. Pour en savoir plus, consultez les ressources suivantes :

- Présentation de la création d'un journal d'activité
- Intégrations et services pris en charge par CloudTrail

- Configuration des notifications Amazon SNS pour CloudTrail
- Réception de fichiers journaux CloudTrail de plusieurs régions et Réception de fichiers journaux CloudTrail de plusieurs comptes

Toutes les GameLift actions Amazon sont enregistrées CloudTrail et documentées dans la référence des GameLift API Amazon. Par exemple`CreateGameSession`, les appels `CreatePlayerSession` et `UpdateGameSession` les actions génèrent des entrées dans les fichiers CloudTrail journaux.

Chaque événement ou entrée de journal contient des informations sur la personne ayant initié la demande. Les informations relatives à l'identité permettent de déterminer les éléments suivants :

- Si la demande a été effectuée avec les informations d'identification utilisateur racine ou AWS Identity and Access Management (IAM).
- Si la demande a été effectuée avec les informations d'identification de sécurité temporaires d'un rôle ou d'un utilisateur fédéré.
- Si la requête a été effectuée par un autre service AWS.

Pour de plus amples informations, veuillez consulter l'élément userIdentity CloudTrail.

Comprendre les entrées du fichier GameLift journal Amazon

Un journal d'activité est une configuration qui permet d'envoyer des événements sous forme de fichiers journaux à un compartiment Simple Storage Service (Amazon S3) que vous spécifiez. Les fichiers journaux CloudTrail contiennent une ou plusieurs entrées de journal. Un événement représente une demande unique provenant de n'importe quelle source et comprend des informations sur l'action demandée, la date et l'heure de l'action, les paramètres de la requête, etc. Les fichiers journaux CloudTrail ne constituent pas une série ordonnée retraçant les appels d'API publics. Ils ne suivent aucun ordre précis.

L'exemple suivant montre une entrée de journal CloudTrail qui illustre les actions `CreateFleet` et `DescribeFleetAttributes`.

```
{
    "Records": [
        {
            "eventVersion": "1.04",
            "userIdentity": {
                "type": "IAMUser",
```

```
            "principalId": "AIDACKCEVSQ6C2EXAMPLE",
            "arn": "arn:aws:iam::111122223333:user/myUserName",
            "accountId": "111122223333",
            "accessKeyId": AKIAIOSFODNN7EXAMPLE",
            "userName": "myUserName"
        },
        "eventTime": "2015-12-29T23:40:15Z",
        "eventSource": "gamelift.amazonaws.com",
        "eventName": "CreateFleet",
        "awsRegion": "us-west-2",
        "sourceIPAddress": "192.0.2.0",
        "userAgent": "[]",
        "requestParameters": {
            "buildId": "build-92b6e8af-37a2-4c10-93bd-4698ea23de8d",
            "eC2InboundPermissions": [
                {
                    "ipRange": "10.24.34.0/23",
                    "fromPort": 1935,
                    "protocol": "TCP",
                    "toPort": 1935
                }
            ],
            "logPaths": [
                "C:\\game\\serverErr.log",
                "C:\\game\\serverOut.log"
            ],
            "eC2InstanceType": "c5.large",
            "serverLaunchPath": "C:\\game\\MyServer.exe",
            "description": "Test fleet",
            "serverLaunchParameters": "-paramX=baz",
            "name": "My_Test_Server_Fleet"
        },
        "responseElements": {
            "fleetAttributes": {
                "fleetId": "fleet-0bb84136-4f69-4bb2-bfec-a9b9a7c3d52e",
                "serverLaunchPath": "C:\\game\\MyServer.exe",
                "status": "NEW",
                "logPaths": [
                    "C:\\game\\serverErr.log",
                    "C:\\game\\serverOut.log"
                ],
                "description": "Test fleet",
                "serverLaunchParameters": "-paramX=baz",
                "creationTime": "Dec 29, 2015 11:40:14 PM",
```

```
                    "name": "My_Test_Server_Fleet",
                    "buildId": "build-92b6e8af-37a2-4c10-93bd-4698ea23de8d"
                }
            },
            "requestID": "824a2a4b-ae85-11e5-a8d6-61d5cafb25f2",
            "eventID": "c8fbea01-fbf9-4c4e-a0fe-ad7dc205ce11",
            "eventType": "AwsApiCall",
            "recipientAccountId": "111122223333"
        },
        {
            "eventVersion": "1.04",
            "userIdentity": {
                "type": "IAMUser",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::111122223333:user/myUserName",
                "accountId": "111122223333",
                "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
                "userName": "myUserName"
            },
            "eventTime": "2015-12-29T23:40:15Z",
            "eventSource": "gamelift.amazonaws.com",
            "eventName": "DescribeFleetAttributes",
            "awsRegion": "us-west-2",
            "sourceIPAddress": "192.0.2.0",
            "userAgent": "[]",
            "requestParameters": {
                "fleetIds": [
                    "fleet-0bb84136-4f69-4bb2-bfec-a9b9a7c3d52e"
                ]
            },
            "responseElements": null,
            "requestID": "82e7f0ec-ae85-11e5-a8d6-61d5cafb25f2",
            "eventID": "11daabcb-0094-49f2-8b3d-3a63c8bad86f",
            "eventType": "AwsApiCall",
            "recipientAccountId": "111122223333"
        },
    ]
}
```

Enregistrement des messages du serveur dans Amazon GameLift

Vous pouvez capturer des messages de serveur personnalisés à partir de vos GameLift serveurs Amazon dans des fichiers journaux. La façon dont vous configurez la journalisation varie selon

que vous utilisez des serveurs personnalisés ou des serveurs en temps réel (voir les sous-sections appropriées dans ce chapitre).

Rubriques

Journalisation des messages du serveur (serveurs personnalisés)

Vous pouvez capturer des messages de serveur personnalisés à partir de vos serveurs GameLift personnalisés Amazon dans des fichiers journaux. Pour en savoir plus sur la journalisation pour les serveurs en temps réel, consultezJournalisation des messages du serveur (serveurs en temps réel).

> **Important**
>
> La taille d'un fichier journal par session de jeu est limitée (voir Amazon GameLift Endpoints and Quotas dans le Références générales AWS). À la fin d'une session de jeu, Amazon GameLift télécharge les journaux du serveur sur Amazon Simple Storage Service (Amazon S3). Amazon ne GameLift téléchargera pas de journaux dépassant cette limite. Les grumes peuvent pousser très rapidement et dépasser la limite de taille. Vous devez surveiller vos journaux et limiter leur sortie aux seuls messages nécessaires.

Configuration de la journalisation pour les serveurs personnalisés

Avec les serveurs GameLift personnalisés Amazon, vous écrivez votre propre code pour effectuer la journalisation, que vous configurez dans le cadre de la configuration du processus de votre serveur. Amazon GameLift utilise votre configuration de journalisation pour identifier les fichiers qu'il doit télécharger sur Amazon S3 à la fin de chaque session de jeu.

Les instructions suivantes montrent comment configurer la journalisation à l'aide d'exemples de code simplifiés :

C++

Pour configurer la journalisation (C++)

1. Créez un vecteur de chaînes qui sont des chemins de répertoire vers les fichiers journaux du serveur de jeu.

```
std::string serverLog("serverOut.log");        // Example server log file
std::vector<std::string> logPaths;
logPaths.push_back(serverLog);
```

2. Indiquez votre vecteur comme étant celui LogParametersde votre ProcessParametersobjet.

```
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. Fournissez l'ProcessParametersobjet lorsque vous appelez ProcessReady().

```
Aws::GameLift::GenericOutcome outcome =
   Aws::GameLift::Server::ProcessReady(processReadyParameter);
```

Pour un exemple plus complet, voirProcessReady().

C#

Pour configurer la journalisation (C#)

1. Créez une liste de chaînes qui sont des chemins de répertoire vers les fichiers journaux du serveur de jeu.

```
List<string> logPaths = new List<string>();
logPaths.Add("C:\\game\\serverOut.txt");     // Example of a log file that the
 game server writes
```

2. Indiquez votre liste en LogParameterstant qu'ProcessParametersobjet.

```
var processReadyParameter = new ProcessParameters(
    this.OnGameSession,
    this.OnProcessTerminate,
    this.OnHealthCheck,
    this.OnGameSessionUpdate,
    port,
```

```
    new LogParameters(logPaths));
```

3. Fournissez l'ProcessParametersobjet lorsque vous appelez ProcessReady().

```
var processReadyOutcome =
   GameLiftServerAPI.ProcessReady(processReadyParameter);
```

Pour un exemple plus complet, voirProcessReady().

Écrire dans des journaux

Vos fichiers journaux existent une fois que le processus de votre serveur a démarré. Vous pouvez écrire dans les journaux en utilisant n'importe quelle méthode pour écrire dans des fichiers. Pour capturer toutes les sorties standard et les sorties d'erreur de votre serveur, remappez les flux de sortie en fichiers journaux, comme dans les exemples suivants :

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"));
```

Accès aux journaux du serveur

À la fin d'une session de jeu, Amazon stocke GameLift automatiquement les journaux dans un compartiment Amazon S3 et les conserve pendant 14 jours. Pour obtenir l'emplacement des journaux d'une session de jeu, vous pouvez utiliser l'opération GetGameSessionLogUrlAPI. Pour télécharger les journaux, utilisez l'URL renvoyée par l'opération.

Journalisation des messages du serveur (serveurs en temps réel)

Vous pouvez capturer des messages de serveur personnalisés à partir de vos serveurs en temps réel dans des fichiers journaux. Pour en savoir plus sur la journalisation pour les serveurs personnalisés, consultezJournalisation des messages du serveur (serveurs personnalisés).

Il existe différents types de messages que vous pouvez enregistrer dans vos fichiers journaux (voirEnregistrement des messages dans le script de votre serveur). Outre vos messages personnalisés, vos serveurs en temps réel émettent des messages système utilisant les mêmes types de messages et écrivent dans les mêmes fichiers journaux. Vous pouvez ajuster le niveau de journalisation de votre parc afin de réduire le nombre de messages de journalisation générés par vos serveurs (voirRéglage du niveau de journalisation).

> **Important**
>
> La taille d'un fichier journal par session de jeu est limitée (voir Amazon GameLift Endpoints and Quotas dans le Références générales AWS). À la fin d'une session de jeu, Amazon GameLift télécharge les journaux du serveur sur Amazon Simple Storage Service (Amazon S3). Amazon ne GameLift téléchargera pas de journaux dépassant cette limite. Les grumes peuvent pousser très rapidement et dépasser la limite de taille. Vous devez surveiller vos journaux et limiter leur sortie aux seuls messages nécessaires.

Enregistrement des messages dans le script de votre serveur

Vous pouvez générer des messages personnalisés dans le script pour vos serveurs en temps réel. Pour envoyer des messages du serveur à un fichier journal, procédez comme suit :

1. Créez une variable pour contenir la référence à l'objet enregistreur.

   ```
   var logger;
   ```

2. Dans la `init()` fonction, récupérez l'enregistreur à partir de l'objet de session et attribuez-le à votre variable d'enregistrement.

   ```
   function init(rtSession) {
       session = rtSession;
       logger = session.getLogger();
   }
   ```

3. Appelez la fonction appropriée de l'enregistreur pour afficher un message.

   Messages de débogage

   ```
   logger.debug("This is my debug message...");
   ```

Messages d'information

```
logger.info("This is my info message...");
```

Messages d'avertissement

```
logger.warn("This is my warn message...");
```

Messages d'erreur

```
logger.error("This is my error message...");
```

Messages d'erreur fatals

```
logger.fatal("This is my fatal error message...");
```

Messages d'erreur fatale liés à l'expérience client

```
logger.cxfatal("This is my customer experience fatal error message...");
```

Pour un exemple des instructions de journalisation dans un script, consultezExemple de script de serveurs en temps réel.

La sortie des fichiers journaux indique le type de message (`DEBUG`,,`INFO`,`WARN`, `ERROR`FATAL,`CXFATAL`), comme le montrent les lignes suivantes d'un exemple de journal :

```
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.
```

Accès aux journaux du serveur

À la fin d'une session de jeu, Amazon stocke GameLift automatiquement les journaux dans Amazon S3 et les conserve pendant 14 jours. Vous pouvez utiliser l'appel d'GetGameSessionLogUrl API pour obtenir l'emplacement des journaux d'une session de jeu. Utilisez l'URL renvoyée par l'appel d'API pour télécharger les journaux.

Réglage du niveau de journalisation

Les grumes peuvent pousser très rapidement et dépasser la limite de taille. Vous devez surveiller vos journaux et limiter leur sortie aux seuls messages nécessaires. Pour les serveurs en temps réel, vous pouvez ajuster le niveau de journalisation en fournissant un paramètre dans la configuration d'exécution de votre flotte sous la forme`loggingLevel:`*`LOGGING_LEVEL`*, où *`LOGGING_LEVEL`* figure l'une des valeurs suivantes :

1. `debug`
2. `info`(par défaut)
3. `warn`
4. `error`
5. `fatal`
6. `cxfatal`

Cette liste est ordonnée du moins grave (`debug`) au plus grave (`cxfatal`). Vous définissez un seul `loggingLevel` et le serveur n'enregistrera que les messages correspondant à ce niveau de gravité ou à un niveau de gravité supérieur. Par exemple, le paramètre `loggingLevel:error` obligera tous les serveurs de votre parc à écrire `error` et à `cxfatal` envoyer des messages uniquement dans le journal. `fatal`

Vous pouvez définir le niveau de journalisation de votre flotte lors de sa création ou après son fonctionnement. La modification du niveau de journalisation de votre flotte après son exécution n'affectera que les journaux des sessions de jeu créées après la mise à jour. Les journaux des sessions de jeu existantes ne seront pas affectés. Si vous ne définissez pas de niveau de journalisation lors de la création de votre flotte, vos serveurs définiront le niveau de journalisation `info` par défaut. Reportez-vous aux sections suivantes pour obtenir des instructions permettant de définir le niveau de journalisation.

Configuration du niveau de journalisation lors de la création d'un parc de serveurs en temps réel (console)

Suivez les instructions de la Créez une flotte GameLift gérée par Amazon section pour créer votre flotte, en ajoutant ce qui suit :

- Dans la sous-étape d'allocation des processus du serveur de l'étape de gestion des processus, fournissez la paire clé-valeur du niveau de journalisation (telle que**`loggingLevel:error`**) en tant que valeur pour les paramètres de lancement. Utilisez un caractère non alphanumérique (sauf une virgule) pour séparer le niveau de journalisation de tout paramètre supplémentaire (par exemple,). `loggingLevel:error +map Winter444`

Configuration du niveau de journalisation lors de la création d'un parc de serveurs en temps réel () AWS CLI

Suivez les instructions de la Créez une flotte GameLift gérée par Amazon section pour créer votre flotte, en ajoutant ce qui suit :

- Dans l'argument du `--runtime-configuration` paramètre for`create-fleet`, indiquez la paire clé-valeur du niveau de journalisation (telle que`loggingLevel:error`) en tant que valeur pour. `Parameters` Utilisez un caractère non alphanumérique (sauf une virgule) pour séparer le niveau de journalisation de tout paramètre supplémentaire. Consultez l'exemple suivant:

```
--runtime-configuration "GameSessionActivationTimeoutSeconds=60,
                    MaxConcurrentGameSessionActivations=2,
                    ServerProcesses=[{LaunchPath=/local/game/myRealtimeLaunchScript.js,
                                    Parameters=loggingLevel:error +map Winter444,
                                    ConcurrentExecutions=10}]"
```

Configuration du niveau de journalisation pour un parc de serveurs en temps réel en cours d'exécution (console)

Suivez les instructions de la section Mettre à jour la configuration d'une flotte pour mettre à jour votre flotte à l'aide de la GameLift console Amazon, en ajoutant ce qui suit :

- Sur la page Modifier le parc, sous Allocation des processus du serveur, indiquez la paire clé-valeur du niveau de journalisation (telle que`loggingLevel:error`) en tant que valeur pour les paramètres de lancement. Utilisez un caractère non alphanumérique (sauf une virgule)

pour séparer le niveau de journalisation de tout paramètre supplémentaire (par exemple,). `loggingLevel:error +map Winter444`

Configuration du niveau de journalisation pour un parc de serveurs en temps réel en cours d'exécution () AWS CLI

Suivez les instructions indiquées Mettre à jour la configuration d'une flotte pour mettre à jour votre flotte à l'aide duAWS CLI, avec l'ajout suivant :

- Dans l'argument du `--runtime-configuration` paramètre for`update-runtime-configuration`, indiquez la paire clé-valeur du niveau de journalisation (telle que`loggingLevel:error`) en tant que valeur pour. `Parameters` Utilisez un caractère non alphanumérique (sauf une virgule) pour séparer le niveau de journalisation de tout paramètre supplémentaire. Consultez l'exemple suivant:

```
--runtime-configuration "GameSessionActivationTimeoutSeconds=60,
                   MaxConcurrentGameSessionActivations=2,
                   ServerProcesses=[{LaunchPath=/local/game/myRealtimeLaunchScript.js,
                                  Parameters=loggingLevel:error +map Winter444,
                                  ConcurrentExecutions=10}]"
```

Sécurité sur Amazon GameLift

Si vous utilisez Amazon GameLift FleetIQ en tant que fonctionnalité autonome avec Amazon EC2, consultez la section Sécurité dans Amazon EC2 dans le guide de l'utilisateur d'Amazon EC2.

La sécurité du cloud AWS est la priorité absolue. En tant que client AWS , vous bénéficiez de centres de données et d'architectures réseau conçus pour répondre aux exigences des organisations les plus pointilleuses en termes de sécurité.

La sécurité est une responsabilité partagée entre vous AWS et vous. Le modèle de responsabilité partagée décrit cela comme la sécurité du cloud et la sécurité dans le cloud :

- Sécurité du cloud : AWS est chargée de protéger l'infrastructure qui exécute les AWS services dans le AWS cloud. AWS vous fournit également des services que vous pouvez utiliser en toute sécurité. Des auditeurs tiers testent et vérifient régulièrement l'efficacité de notre sécurité dans le AWS cadre des programmes de de). Pour en savoir plus sur les programmes de conformité qui s'appliquent à Amazon GameLift, consultez la section AWS services concernés par programme de conformité et .
- Sécurité dans le cloud — Votre responsabilité est déterminée par le AWS service que vous utilisez. Vous êtes également responsable d'autres facteurs, notamment la sensibilité de vos données, les exigences de votre entreprise et les lois AWS et réglementations applicables.

Cette documentation vous aide à comprendre comment appliquer le modèle de responsabilité partagée lors de l'utilisation d'Amazon GameLift. Les rubriques suivantes expliquent comment configurer Amazon pour répondre GameLift à vos objectifs de sécurité et de conformité. Vous apprendrez également à utiliser d'autres AWS services qui vous aident à surveiller et à sécuriser vos GameLift ressources Amazon.

Rubriques

- Analyse de configuration et de vulnérabilité sur Amazon GameLift
- Bonnes pratiques de sécurité pour Amazon GameLift

Protection des données sur Amazon GameLift

Si vous utilisez Amazon GameLift FleetIQ en tant que fonctionnalité autonome avec Amazon EC2, consultez la section Sécurité dans Amazon EC2 dans le guide de l'utilisateur d'Amazon EC2.

Le modèle de responsabilité AWS partagée de s'applique à la protection des données sur Amazon GameLift. Comme décrit dans ce modèle, AWS est chargé de protéger l'infrastructure mondiale qui gère tous les AWS Cloud. La gestion du contrôle de votre contenu hébergé sur cette infrastructure relève de votre responsabilité. Vous êtes également responsable des tâches de configuration et de gestion de la sécurité des Services AWS que vous utilisez. Pour plus d'informations sur la confidentialité des données, consultez Questions fréquentes (FAQ) sur la confidentialité des données. Pour en savoir plus sur la protection des données en Europe, consultez le billet de blog Modèle de responsabilité partagée AWS et RGPD (Règlement général sur la protection des données) sur le Blog de sécuritéAWS .

À des fins de protection des données, nous vous recommandons de protéger les Compte AWS informations d'identification et de configurer les utilisateurs individuels avec AWS IAM Identity Center ou AWS Identity and Access Management (IAM). Ainsi, chaque utilisateur se voit attribuer uniquement les autorisations nécessaires pour exécuter ses tâches. Nous vous recommandons également de sécuriser vos données comme indiqué ci-dessous :

- Utilisez l'authentification multifactorielle (MFA) avec chaque compte.
- Utilisez le protocole SSL/TLS pour communiquer avec les ressources. AWS Nous exigeons TLS 1.2 et recommandons TLS 1.3.
- Configurez l'API et la journalisation de l'activité des utilisateurs avec AWS CloudTrail.
- Utilisez des solutions de AWS chiffrement, ainsi que tous les contrôles de sécurité par défaut qu'ils contiennent Services AWS.
- Utilisez des services de sécurité gérés avancés tels qu'Amazon Macie, qui contribuent à la découverte et à la sécurisation des données sensibles stockées dans Amazon S3.
- Si vous avez besoin de modules cryptographiques validés par la norme FIPS 140-2 pour accéder AWS via une interface de ligne de commande ou une API, utilisez un point de terminaison FIPS. Pour plus d'informations sur les points de terminaison FIPS (Federal Information Processing

Standard) disponibles, consultez Federal Information Processing Standard (FIPS) 140-2 (Normes de traitement de l'information fédérale).

Nous vous recommandons fortement de ne jamais placer d'informations confidentielles ou sensibles, telles que les adresses e-mail de vos clients, dans des balises ou des champs de texte libre tels que le champ Name (Nom). Cela inclut lorsque vous travaillez avec Amazon GameLift ou une autre entreprise Services AWS à l'aide de la console, de l'API ou AWS des SDK. AWS CLI Toutes les données que vous entrez dans des balises ou des champs de texte de forme libre utilisés pour les noms peuvent être utilisées à des fins de facturation ou dans les journaux de diagnostic. Si vous fournissez une adresse URL à un serveur externe, nous vous recommandons fortement de ne pas inclure d'informations d'identification dans l'adresse URL permettant de valider votre demande adressée à ce serveur.

Les données GameLift spécifiques à Amazon sont traitées comme suit :

- Les versions de serveurs de jeu et les scripts que vous chargez sur Amazon GameLift sont stockés dans Amazon S3. Il n'y a pas d'accès client direct à ces données une fois qu'elles sont chargées. Un utilisateur autorisé peut obtenir un accès temporaire pour charger des fichiers, mais ne peut pas afficher ou mettre à jour les fichiers directement dans Amazon S3. Pour supprimer des scripts et des builds, utilisez la GameLift console Amazon ou l'API du service.
- Les données du journal de session de jeu sont stockées dans Amazon S3 pendant une période limitée une fois la session de jeu terminée. Les utilisateurs autorisés peuvent accéder aux données du journal en les téléchargeant via un lien dans la GameLift console Amazon ou en appelant l'API du service.
- Les données relatives aux mesures et aux événements sont stockées sur Amazon GameLift et sont accessibles via la GameLift console Amazon ou en appelant l'API du service. Les données pouvant être récupérées concernent les parcs, les instances, les placements de session de jeu, les tickets de mise en relation, les sessions de jeu et les sessions de joueur. Les données sont également accessibles via Amazon CloudWatch et CloudWatch Events.
- Les données fournies par le client sont stockées sur Amazon. GameLift Les utilisateurs autorisés peuvent y accéder via des appels à l'API du service. Les données potentiellement sensibles peuvent inclure des données de joueur, des données de session de joueur et de session de jeu (y compris les informations de connexion), des données de mise en relation, etc.

> **Note**
>
> Si vous fournissez des ID de joueur personnalisés dans vos demandes, ces valeurs doivent prendre la forme d'UUID anonymisés et ne contenir aucune information d'identification de joueur.

Pour de plus amples informations sur la protection des données, veuillez consulter le billet de blog Modèle de responsabilité partagée AWS et RGPD à la page Blog sur la sécuritéAWS .

Chiffrement au repos

Le chiffrement au repos des données GameLift spécifiques à Amazon est géré comme suit :

- Les versions et les scripts des serveurs de jeu sont stockés dans des compartiments Amazon S3 avec chiffrement côté serveur.
- Les données fournies par le client sont stockées sur Amazon GameLift dans un format crypté.

Chiffrement en transit

Les connexions aux GameLift API Amazon sont établies via une connexion sécurisée (SSL) et authentifiées à l'aide de AWS Signature Version 4 (lors de la connexion via la AWS CLI ou le AWS SDK, la signature est gérée automatiquement). L'authentification est gérée à l'aide des stratégies d'accès définies par IAM pour les informations d'identification de sécurité utilisées pour établir la connexion.

La communication directe entre les clients de jeu et les serveurs de jeu est la suivante :

- Pour les serveurs de jeux personnalisés hébergés sur les GameLift ressources Amazon, la communication n'implique pas le GameLift service Amazon. Le chiffrement de cette communication est de la responsabilité du client. Vous pouvez utiliser des parcs compatibles TLS pour que vos clients de jeu authentifient le serveur de jeu lors de la connexion et chiffrent toutes les communications entre votre client de jeu et votre serveur de jeu.
- Pour les serveurs en temps réel sur lesquels la génération de certificats TLS est activée, le trafic entre le client de jeu et les serveurs en temps réel utilisant le SDK du client en temps réel est crypté en cours de vol. Le trafic TCP est chiffré avec TLS 1.2 et le trafic UDP avec DTLS 1.2.

Confidentialité du trafic inter-réseau

Vous pouvez accéder à distance à vos GameLift instances Amazon en toute sécurité. Pour les instances qui utilisent Linux, SSH fournit un canal de communication sécurisé pour l'accès à distance. Pour les instances qui exécutent Windows, utilisez un client RDP (Remote Desktop Protocol). Avec Amazon GameLift FleetIQ, l'accès à distance à vos instances à l'aide de Systems AWS Manager Session Manager et Run Command est chiffré à l'aide du protocole TLS 1.2, et les demandes de création de connexion sont signées à l'aide de Sigv4. Pour obtenir de l'aide sur la connexion à une GameLift instance Amazon gérée, consultezConnectez-vous à distance aux instances GameLift de flotte Amazon.

Gestion des identités et des accès pour Amazon GameLift

AWS Identity and Access Management (IAM) est un Service AWS qui aide un administrateur à contrôler en toute sécurité l'accès aux ressources AWS. Les administrateurs IAM contrôlent qui peut être authentifié (connecté) et autorisé (autorisé) à utiliser les ressources Amazon GameLift . IAM est un Service AWS que vous pouvez utiliser sans frais supplémentaires.

Rubriques

Public ciblé

La façon dont vous utilisez AWS Identity and Access Management (IAM) varie en fonction du travail que vous effectuez sur Amazon GameLift.

Utilisateur du service — Si vous utilisez le GameLift service Amazon pour effectuer votre travail, votre administrateur vous fournit les informations d'identification et les autorisations dont vous avez besoin. Au fur et à mesure que vous utilisez de plus en plus de GameLift fonctionnalités Amazon pour effectuer votre travail, vous aurez peut-être besoin d'autorisations supplémentaires.

En comprenant bien la gestion des accès, vous saurez demander les autorisations appropriées à votre administrateur. Si vous ne parvenez pas à accéder à une fonctionnalité d'Amazon GameLift, consultezRésolution des problèmes d' GameLift identité et d'accès à Amazon.

Administrateur du service — Si vous êtes responsable des GameLift ressources Amazon au sein de votre entreprise, vous avez probablement un accès complet à Amazon GameLift. C'est à vous de déterminer à quelles GameLift fonctionnalités et ressources Amazon les utilisateurs de vos services doivent accéder. Vous devez ensuite soumettre les demandes à votre administrateur IAM pour modifier les autorisations des utilisateurs de votre service. Consultez les informations sur cette page pour comprendre les concepts de base d'IAM. Pour en savoir plus sur la manière dont votre entreprise peut utiliser IAM avec Amazon GameLift, consultezComment Amazon GameLift travaille avec IAM.

Administrateur IAM — Si vous êtes administrateur IAM, vous souhaiterez peut-être en savoir plus sur la manière dont vous pouvez rédiger des politiques pour gérer l'accès à Amazon. GameLift Pour consulter des exemples de politiques GameLift basées sur l'identité Amazon que vous pouvez utiliser dans IAM, consultez. Exemples de politiques basées sur l'identité pour Amazon GameLift

Authentification par des identités

L'authentification correspond au processus par lequel vous vous connectez à AWS avec vos informations d'identification. Vous devez vous authentifier (être connecté à AWS) en tant qu'Utilisateur racine d'un compte AWS, en tant qu'utilisateur IAM ou en endossant un rôle IAM.

Vous pouvez vous connecter à AWS en tant qu'identité fédérée à l'aide des informations d'identification fournies par le biais d'une source d'identité. AWS IAM Identity Center Les utilisateurs (IAM Identity Center), l'authentification de connexion unique de votre entreprise et vos informations d'identification Google ou Facebook sont des exemples d'identités fédérées. Lorsque vous vous connectez avec une identité fédérée, votre administrateur aura précédemment configuré une fédération d'identités avec des rôles IAM. Lorsque vous accédez à AWS en utilisant la fédération, vous endossez indirectement un rôle.

Selon le type d'utilisateur que vous êtes, vous pouvez vous connecter à la AWS Management Console ou au portail d'accès AWS. Pour plus d'informations sur la connexion à AWS, consultez Connexion à votre Compte AWSdans le Guide de l'utilisateurConnexion à AWS.

Si vous accédez à AWS par programmation, AWS fournit un kit de développement logiciel (SDK) et une interface de ligne de commande (CLI) pour signer cryptographiquement vos demandes en utilisant vos informations d'identification. Si vous n'utilisez pas les outils AWS, vous devez signer les

requêtes vous-même. Pour plus d'informations sur l'utilisation de la méthode recommandée pour signer des demandes vous-même, consultez Signature des demandes d'API AWS dans le Guide de l'utilisateur IAM.

Quelle que soit la méthode d'authentification que vous utilisez, vous devrez peut-être fournir des informations de sécurité supplémentaires. Par exemple, AWS vous recommande d'utiliser l'authentification multifactorielle (MFA) pour améliorer la sécurité de votre compte. Pour en savoir plus, veuillez consulter Authentification multifactorielle dans le Guide de l'utilisateur AWS IAM Identity Center et Utilisation de l'authentification multifactorielle (MFA) dans l'interface AWS dans le Guide de l'utilisateur IAM.

Utilisateur root Compte AWS

Lorsque vous créez un Compte AWS, vous commencez avec une seule identité de connexion disposant d'un accès complet à tous les Services AWS et ressources du compte. Cette identité est appelée utilisateur root du Compte AWS. Vous pouvez y accéder en vous connectant à l'aide de l'adresse électronique et du mot de passe que vous avez utilisés pour créer le compte. Il est vivement recommandé de ne pas utiliser l'utilisateur root pour vos tâches quotidiennes. Protégez vos informations d'identification d'utilisateur root et utilisez-les pour effectuer les tâches que seul l'utilisateur root peut effectuer. Pour obtenir la liste complète des tâches qui vous imposent de vous connecter en tant qu'utilisateur root, consultez Tâches nécessitant des informations d'identification d'utilisateur root dans le Guide de l'utilisateur IAM.

Identité fédérée

Demandez aux utilisateurs humains, et notamment aux utilisateurs qui nécessitent un accès administrateur, d'appliquer la bonne pratique consistant à utiliser une fédération avec fournisseur d'identité pour accéder à Services AWS en utilisant des informations d'identification temporaires.

Une identité fédérée est un utilisateur de l'annuaire des utilisateurs de votre entreprise, un fournisseur d'identité Web, l'AWS Directory Service, l'annuaire Identity Center ou tout utilisateur qui accède à Services AWS en utilisant des informations d'identification fournies via une source d'identité. Quand des identités fédérées accèdent à Comptes AWS, elles endossent des rôles, ces derniers fournissant des informations d'identification temporaires.

Pour une gestion des accès centralisée, nous vous recommandons d'utiliser AWS IAM Identity Center. Vous pouvez créer des utilisateurs et des groupes dans IAM Identity Center, ou vous connecter et vous synchroniser avec un ensemble d'utilisateurs et de groupes dans votre propre source d'identité pour une utilisation sur l'ensemble de vos applications et de vos Comptes AWS.

Pour obtenir des informations sur IAM Identity Center, consultez Qu'est-ce que IAM Identity Center ? dans le Guide de l'utilisateur AWS IAM Identity Center.

Utilisateurs et groupes IAM

Un utilisateur IAM est une identité dans votre Compte AWS qui dispose d'autorisations spécifiques pour une seule personne ou application. Dans la mesure du possible, nous vous recommandons de vous appuyer sur des informations d'identification temporaires plutôt que de créer des utilisateurs IAM ayant des informations d'identification à long terme tels que les clés d'accès. Toutefois, si certains cas d'utilisation spécifiques nécessitent des informations d'identification à long terme avec les utilisateurs IAM, nous vous recommandons de faire pivoter les clés d'accès. Pour plus d'informations, consultez Rotation régulière des clés d'accès pour les cas d'utilisation nécessitant des informations d'identification dans le Guide de l'utilisateur IAM.

Un groupe IAM est une identité qui concerne un ensemble d'utilisateurs IAM. Vous ne pouvez pas vous connecter en tant que groupe. Vous pouvez utiliser les groupes pour spécifier des autorisations pour plusieurs utilisateurs à la fois. Les groupes permettent de gérer plus facilement les autorisations pour de grands ensembles d'utilisateurs. Par exemple, vous pouvez avoir un groupe nommé IAMAdmins et accorder à ce groupe les autorisations d'administrer des ressources IAM.

Les utilisateurs sont différents des rôles. Un utilisateur est associé de manière unique à une personne ou une application, alors qu'un rôle est conçu pour être endossé par tout utilisateur qui en a besoin. Les utilisateurs disposent d'informations d'identification permanentes, mais les rôles fournissent des informations d'identification temporaires. Pour en savoir plus, consultez Quand créer un utilisateur IAM (au lieu d'un rôle) dans le Guide de l'utilisateur IAM.

Rôles IAM

Un rôle IAM est une entité au sein de votre Compte AWS qui dispose d'autorisations spécifiques. Le concept ressemble à celui d'utilisateur IAM, mais le rôle IAM n'est pas associé à une personne en particulier. Vous pouvez temporairement endosser un rôle IAM dans la AWS Management Console en changeant de rôle. Vous pouvez obtenir un rôle en appelant une opération d'API AWS CLI ou AWS à l'aide d'une URL personnalisée. Pour plus d'informations sur les méthodes d'utilisation des rôles, consultez Utilisation de rôles IAM dans le Guide de l'utilisateur IAM.

Les rôles IAM avec des informations d'identification temporaires sont utiles dans les cas suivants :

- Accès utilisateur fédéré – Pour attribuer des autorisations à une identité fédérée, vous créez un rôle et définissez des autorisations pour le rôle. Quand une identité externe s'authentifie, l'identité est associée au rôle et reçoit les autorisations qui sont définies par celui-ci. Pour

obtenir des informations sur les rôles pour la fédération, consultez Création d'un rôle pour un fournisseur d'identité tiers (fédération) dans le Guide de l'utilisateur IAM. Si vous utilisez IAM Identity Center, vous configurez un jeu d'autorisations. IAM Identity Center met en corrélation le jeu d'autorisations avec un rôle dans IAM afin de contrôler à quoi vos identités peuvent accéder après leur authentification. Pour plus d'informations sur les jeux d'autorisations, consultez Jeux d'autorisations dans le Guide de l'utilisateur AWS IAM Identity Center.

- Autorisations d'utilisateur IAM temporaires : un rôle ou un utilisateur IAM peut endosser un rôle IAM pour profiter temporairement d'autorisations différentes pour une tâche spécifique.
- Accès intercompte : vous pouvez utiliser un rôle IAM pour permettre à un utilisateur (principal de confiance) d'un compte différent d'accéder aux ressources de votre compte. Les rôles constituent le principal moyen d'accorder l'accès intercompte. Toutefois, certains Services AWS vous permettent d'attacher une politique directement à une ressource (au lieu d'utiliser un rôle en tant que proxy). Pour en savoir plus sur la différence entre les rôles et les politiques basées sur les ressources pour l'accès intercompte, consultez Différence entre les rôles IAM et les politiques basées sur les ressources dans le Guide de l'utilisateur IAM.
- Accès interservices : certains Services AWS utilisent des fonctions dans d'autres Services AWS. Par exemple, lorsque vous effectuez un appel dans un service, il est courant que ce service exécute des applications dans Amazon EC2 ou stocke des objets dans Amazon S3. Un service peut le faire en utilisant les autorisations d'appel du principal, une fonction du service ou un rôle lié au service.
  - Forward access sessions (FAS) – Lorsque vous utilisez un utilisateur ou un rôle IAM pour effectuer des actions dans AWS, vous êtes considéré comme un principal. Lorsque vous utilisez certains services, vous pouvez effectuer une action qui déclenche une autre action dans un autre service. FAS utilise les autorisations du principal appelant Service AWS, combinées à la demande Service AWS pour effectuer des demandes aux services en aval. Les demandes FAS ne sont formulées que lorsqu'un service reçoit une demande qui, pour aboutir, a besoin d'interagir avec d'autres ressources ou Services AWS. Dans ce cas, vous devez disposer d'autorisations nécessaires pour effectuer les deux actions. Pour plus de détails sur la politique relative à la transmission de demandes FAS, consultez Sessions de transmission d'accès.
  - Fonction du service : il s'agit d'un rôle IAM attribué à un service afin de réaliser des actions en votre nom. Un administrateur IAM peut créer, modifier et supprimer une fonction de service à partir d'IAM. Pour plus d'informations, consultez Création d'un rôle pour la délégation d'autorisations à un Service AWS dans le Guide de l'utilisateur IAM.
  - Rôle lié au service : un rôle lié au service est un type de fonction de service lié à un Service AWS. Le service peut endosser le rôle afin d'effectuer une action en votre nom. Les rôles liés à

un service s'affichent dans votre Compte AWS et sont détenus par le service. Un administrateur IAM peut consulter, mais ne peut pas modifier, les autorisations concernant les rôles liés à un service.

- Applications s'exécutant sur Amazon EC2 : vous pouvez utiliser un rôle IAM pour gérer des informations d'identification temporaires pour les applications s'exécutant sur une instance EC2 et effectuant des demandes d'API AWS CLI ou AWS. Cette solution est préférable au stockage des clés d'accès au sein de l'instance EC2. Pour attribuer un rôle AWS à une instance EC2 et le rendre disponible à toutes les applications associées, vous pouvez créer un profil d'instance attaché à l'instance. Un profil d'instance contient le rôle et permet aux programmes qui s'exécutent sur l'instance EC2 d'obtenir des informations d'identification temporaires. Pour plus d'informations, consultez Utilisation d'un rôle IAM pour accorder des autorisations à des applications s'exécutant sur des instances Amazon EC2 dans le Guide de l'utilisateur IAM.

Pour savoir dans quel cas utiliser des rôles ou des utilisateurs IAM, consultez Quand créer un rôle IAM (au lieu d'un utilisateur) dans le Guide de l'utilisateur IAM.

Gestion des accès à l'aide de politiques

Vous contrôlez les accès dans AWS en créant des politiques et en les attachant à des identités AWS ou à des ressources. Une politique est un objet dans AWS qui, lorsqu'il est associé à une identité ou à une ressource, définit les autorisations de ces dernières. AWS évalue ces politiques lorsqu'un principal (utilisateur, utilisateur racine ou séance de rôle) envoie une demande. Les autorisations dans les politiques déterminent si la demande est autorisée ou refusée. La plupart des politiques sont stockées dans AWS en tant que documents JSON. Pour plus d'informations sur la structure et le contenu des déclarations de politique JSON, consultez Présentation des politiques JSON dans le Guide de l'utilisateur IAM.

Les administrateurs peuvent utiliser les politiques JSON AWS pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

Par défaut, les utilisateurs et les rôles ne disposent d'aucune autorisation. Pour octroyer aux utilisateurs des autorisations d'effectuer des actions sur les ressources dont ils ont besoin, un administrateur IAM peut créer des politiques IAM. L'administrateur peut ensuite ajouter les politiques IAM aux rôles et les utilisateurs peuvent endosser les rôles.

Les politiques IAM définissent les autorisations d'une action, quelle que soit la méthode que vous utilisez pour exécuter l'opération. Par exemple, supposons que vous disposiez d'une politique qui

autorise l'action `iam:GetRole`. Un utilisateur avec cette politique peut obtenir des informations utilisateur à partir de la AWS Management Console, de la AWS CLI ou de l'API AWS.

Politiques basées sur l'identité

Les politiques basées sur l'identité sont des déclarations de politique d'autorisations JSON que vous pouvez attacher à une identité telle qu'un utilisateur, un groupe d'utilisateurs ou un rôle IAM. Ces politiques contrôlent quel type d'actions des utilisateurs et des rôles peuvent exécuter, sur quelles ressources et dans quelles conditions. Pour découvrir comment créer une politique basée sur l'identité, consultez Création de politiques IAM dans le Guide de l'utilisateur IAM.

Les politiques basées sur l'identité peuvent être classées comme des politiques en ligne ou des politiques gérées. Les politiques en ligne sont intégrées directement à un utilisateur, groupe ou rôle. Les politiques gérées sont des politiques autonomes que vous pouvez attacher à plusieurs utilisateurs, groupes et rôles dans votre Compte AWS. Les politiques gérées incluent les politiques gérées par AWS et les politiques gérées par le client. Pour découvrir comment choisir entre une politique gérée et une politique en ligne, consultez Choix entre les politiques gérées et les politiques en ligne dans le Guide de l'utilisateur IAM.

politiques basées sur les ressources

Les politiques basées sur les ressources sont des documents de politique JSON que vous attachez à une ressource. Des politiques basées sur les ressources sont, par exemple, les politiques de confiance de rôle IAM et des politiques de compartiment Amazon S3. Dans les services qui sont compatibles avec les politiques basées sur les ressources, les administrateurs de service peuvent les utiliser pour contrôler l'accès à une ressource spécifique. Pour la ressource dans laquelle se trouve la politique, cette dernière définit quel type d'actions un principal spécifié peut effectuer sur cette ressource et dans quelles conditions. Vous devez spécifier un principal dans une politique basée sur les ressources. Les principaux peuvent inclure des comptes, des utilisateurs, des rôles, des utilisateurs fédérés ou des Services AWS.

Les politiques basées sur les ressources sont des politiques en ligne situées dans ce service. Vous ne pouvez pas utiliser les politiques gérées AWS depuis IAM dans une politique basée sur une ressource.

Listes de contrôle d'accès (ACL)

Les listes de contrôle d'accès (ACL) vérifie quels principaux (membres de compte, utilisateurs ou rôles) ont l'autorisation d'accéder à une ressource. Les listes de contrôle d'accès sont similaires aux

politiques basées sur les ressources, bien qu'elles n'utilisent pas le format de document de politique JSON.

Amazon S3, AWS WAF et Amazon VPC sont des exemples de services prenant en charge les ACL. Pour en savoir plus sur les listes de contrôle d'accès, consultez Présentation des listes de contrôle d'accès (ACL) dans le Guide du développeur Amazon Simple Storage Service.

Autres types de politique

AWS prend en charge d'autres types de politiques moins courantes. Ces types de politiques peuvent définir le nombre maximum d'autorisations qui vous sont accordées par des types de politiques plus courants.

- Limite d'autorisations : une limite d'autorisations est une fonction avancée dans laquelle vous définissez le nombre maximal d'autorisations qu'une politique basée sur l'identité peut accorder à une entité IAM (utilisateur ou rôle IAM). Vous pouvez définir une limite d'autorisations pour une entité. Les autorisations qui en résultent représentent la combinaison des politiques basées sur l'identité d'une entité et de ses limites d'autorisation. Les politiques basées sur les ressources qui spécifient l'utilisateur ou le rôle dans le champ `Principal` ne sont pas limitées par les limites d'autorisations. Un refus explicite dans l'une de ces politiques remplace l'autorisation. Pour plus d'informations sur les limites d'autorisations, consultez Limites d'autorisations pour des entités IAM dans le Guide de l'utilisateur IAM.
- Politiques de contrôle des services (SCP) - les SCP sont des politiques JSON qui spécifient le nombre maximal d'autorisations pour une organisation ou une unité d'organisation (OU) dans AWS Organizations. AWS Organizations est un service qui vous permet de regrouper et de gérer de façon centralisée plusieurs Comptes AWS détenus par votre entreprise. Si vous activez toutes les fonctions d'une organisation, vous pouvez appliquer les politiques de contrôle des services (SCP) à l'un ou à l'ensemble de vos comptes. La SCP limite les autorisations pour les entités dans les comptes membres, y compris dans chaque Utilisateur racine d'un compte AWS. Pour plus d'informations sur les organisations et les SCP, consultez Fonctionnement des SCP dans le Guide de l'utilisateur AWS Organizations.
- politiques de séance : les politiques de séance sont des politiques avancées que vous utilisez en tant que paramètre lorsque vous créez par programmation une séance temporaire pour un rôle ou un utilisateur fédéré. Les autorisations de la séance obtenue sont une combinaison des politiques basées sur l'identité de l'utilisateur ou du rôle et des politiques de séance. Les autorisations peuvent également provenir d'une politique basée sur les ressources. Un refus explicite dans l'une de ces politiques remplace l'autorisation. Pour plus d'informations, consultez Politiques de séance dans le Guide de l'utilisateur IAM.

Plusieurs types de politique

Lorsque plusieurs types de politiques s'appliquent à la requête, les autorisations obtenues sont plus compliquées à comprendre. Pour découvrir la façon dont AWS détermine s'il convient d'autoriser une demande en présence de plusieurs types de politiques, veuillez consulter Logique d'évaluation de politiques dans le Guide de l'utilisateur IAM.

Comment Amazon GameLift travaille avec IAM

Avant d'utiliser IAM pour gérer l'accès à Amazon GameLift, découvrez quelles fonctionnalités IAM peuvent être utilisées avec Amazon. GameLift

Fonctionnalités IAM que vous pouvez utiliser avec Amazon GameLift

Fonctionnalité IAM	GameLift Assistance Amazon
Politiques basées sur l'identité	Oui
Politiques basées sur les ressources	Non
Actions de politique	Oui
Ressources de politique	Oui
Clés de condition de politique (spécifiques au service)	Oui
ACL	Non
ABAC (étiquettes dans les politiques)	Oui
Informations d'identification temporaires	Oui
Autorisations de principal	Oui
Fonctions du service	Oui
Rôles liés à un service	Non

Pour obtenir une vue d'ensemble de la façon dont Amazon GameLift et les autres AWS services fonctionnent avec la plupart des fonctionnalités IAM, consultez les AWSservices compatibles avec IAM dans le guide de l'utilisateur IAM.

Politiques basées sur l'identité pour Amazon GameLift

Prend en charge les politiques basées sur une identité	Oui

Les politiques basées sur l’identité sont des documents de politique d’autorisations JSON que vous pouvez attacher à une identité telle qu’un utilisateur, un Groupes d’utilisateurs IAM ou un rôle IAM. Ces politiques contrôlent quel type d’actions des utilisateurs et des rôles peuvent exécuter, sur quelles ressources et dans quelles conditions. Pour découvrir comment créer une politique basée sur l'identité, consultez Création de politiques IAM dans le Guide de l’utilisateur IAM.

Avec les politiques IAM basées sur l’identité, vous pouvez spécifier des actions et ressources autorisées ou refusées, ainsi que les conditions dans lesquelles les actions sont autorisées ou refusées. Vous ne pouvez pas spécifier le principal dans une politique basée sur une identité car celle-ci s’applique à l’utilisateur ou au rôle auquel elle est attachée. Pour découvrir tous les éléments que vous utilisez dans une politique JSON, consultez Références des éléments de politique JSON IAM dans le Guide de l’utilisateur IAM.

Exemples de politiques basées sur l'identité pour Amazon GameLift

Pour consulter des exemples de politiques GameLift basées sur l'identité d'Amazon, consultez. Exemples de politiques basées sur l'identité pour Amazon GameLift

Politiques basées sur les ressources au sein d'Amazon GameLift

Prend en charge les politiques basées sur une ressource	Non

Les politiques basées sur les ressources sont des documents de politique JSON que vous attachez à une ressource. Des politiques basées sur les ressources sont, par exemple, les politiques de confiance de rôle IAM et des politiques de compartiment Amazon S3. Dans les services qui sont compatibles avec les politiques basées sur les ressources, les administrateurs de service peuvent les

utiliser pour contrôler l'accès à une ressource spécifique. Pour la ressource dans laquelle se trouve la politique, cette dernière définit quel type d'actions un principal spécifié peut effectuer sur cette ressource et dans quelles conditions. Vous devez spécifier un principal dans une politique basée sur les ressources. Les principaux peuvent inclure des comptes, des utilisateurs, des rôles, des utilisateurs fédérés ou des Services AWS.

Pour permettre un accès intercompte, vous pouvez spécifier un compte entier ou des entités IAM dans un autre compte en tant que principal dans une politique basée sur les ressources. L'ajout d'un principal entre comptes à une politique basée sur les ressources ne représente qu'une partie de l'instauration de la relation d'approbation. Quand le principal et la ressource se trouvent dans des Comptes AWS différents, un administrateur IAM dans le compte approuvé doit également accorder à l'entité principal (utilisateur ou rôle) l'autorisation d'accéder à la ressource. Pour ce faire, il attache une politique basée sur une identité à l'entité. Toutefois, si une politique basée sur des ressources accorde l'accès à un principal dans le même compte, aucune autre politique basée sur l'identité n'est requise. Pour plus d'informations, consultez Différence entre les rôles IAM et les politiques basées sur une ressource dans le Guide de l'utilisateur IAM.

Actions politiques pour Amazon GameLift

Prend en charge les actions de politique	Oui

Les administrateurs peuvent utiliser les politiques JSON AWS pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément `Action` d'une politique JSON décrit les actions que vous pouvez utiliser pour autoriser ou refuser l'accès à une politique. Les actions de politique possèdent généralement le même nom que l'opération d'API AWS associée. Il existe quelques exceptions, telles que les actions avec autorisations uniquement qui n'ont pas d'opération API correspondante. Certaines opérations nécessitent également plusieurs actions dans une politique. Ces actions supplémentaires sont nommées actions dépendantes.

Intégration d'actions dans une stratégie afin d'accorder l'autorisation d'exécuter les opérations associées.

Pour obtenir la liste des GameLift actions Amazon, consultez la section Actions définies par Amazon GameLift dans le Service Authorization Reference.

Les actions politiques sur Amazon GameLift utilisent le préfixe suivant avant l'action :

```
gamelift
```

Pour indiquer plusieurs actions dans une seule déclaration, séparez-les par des virgules.

```
"Action": [
      "gamelift:action1",
      "gamelift:action2"
         ]
```

Vous pouvez aussi spécifier plusieurs actions à l'aide de caractères génériques (*). Par exemple, pour spécifier toutes les actions qui commencent par le mot `Describe`, incluez l'action suivante :

```
"Action": "gamelift:Describe*"
```

Pour consulter des exemples de politiques GameLift basées sur l'identité d'Amazon, consultez. Exemples de politiques basées sur l'identité pour Amazon GameLift

Ressources relatives aux politiques pour Amazon GameLift

Prend en charge les ressources de politique	Oui

Les administrateurs peuvent utiliser les politiques JSON AWS pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément de politique JSON `Resource` indique le ou les objets pour lesquels l'action s'applique. Les instructions doivent inclure un élément `Resource` ou `NotResource`. Il est recommandé de définir une ressource à l'aide de son Amazon Resource Name (ARN). Vous pouvez le faire pour des actions qui prennent en charge un type de ressource spécifique, connu sous la dénomination autorisations de niveau ressource.

Pour les actions qui ne sont pas compatibles avec les autorisations de niveau ressource, telles que les opérations de liste, utilisez un caractère générique (*) afin d'indiquer que l'instruction s'applique à toutes les ressources.

```
"Resource": "*"
```

Pour obtenir la liste des types de GameLift ressources Amazon et de leurs ARN, consultez la section Ressources définies par Amazon GameLift dans le Service Authorization Reference. Pour savoir avec quelles actions vous pouvez spécifier l'ARN de chaque ressource, consultez Actions définies par Amazon GameLift.

Certaines GameLift ressources Amazon ont des valeurs ARN, ce qui permet de gérer leur accès à l'aide de politiques IAM. La ressource GameLift de flotte Amazon possède un ARN dont la syntaxe est la suivante :

```
arn:${Partition}:gamelift:${Region}:${Account}:fleet/${FleetId}
```

Pour de plus amples informations sur le format des ARN, veuillez consulter Amazon Ressource Names (ARN) dans la Références générales AWS.

Par exemple, pour spécifier le parc `fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa` dans votre déclaration, utilisez l'ARN suivant :

```
"Resource": "arn:aws:gamelift:us-west-2:123456789012:fleet/
fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa"
```

Pour spécifier toutes les flottes appartenant à un compte spécifique, utilisez un caractère générique (*) :

```
"Resource": "arn:aws:gamelift:us-west-2:123456789012:fleet/*"
```

Pour consulter des exemples de politiques GameLift basées sur l'identité d'Amazon, consultez. Exemples de politiques basées sur l'identité pour Amazon GameLift

Clés relatives aux conditions de politique pour Amazon GameLift

Prise en charge des clés de condition de stratégie spécifiques au service	Oui

Les administrateurs peuvent utiliser les politiques JSON AWS pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément `Condition` (ou le bloc `Condition`) vous permet de spécifier des conditions lorsqu'une instruction est appliquée. L'élément `Condition` est facultatif. Vous pouvez créer des expressions conditionnelles qui utilisent des opérateurs de condition, tels que les signes égal ou inférieur à, pour faire correspondre la condition de la politique aux valeurs de la demande.

Si vous spécifiez plusieurs éléments `Condition` dans une instruction, ou plusieurs clés dans un seul élément `Condition`, AWS les évalue à l'aide d'une opération `AND` logique. Si vous spécifiez plusieurs valeurs pour une seule clé de condition, AWS évalue la condition à l'aide d'une opération `OR` logique. Toutes les conditions doivent être remplies avant que les autorisations associées à l'instruction ne soient accordées.

Vous pouvez aussi utiliser des variables d'espace réservé quand vous spécifiez des conditions. Par exemple, vous pouvez accorder à un utilisateur IAM l'autorisation d'accéder à une ressource uniquement si elle est balisée avec son nom d'utilisateur IAM. Pour plus d'informations, consultez Éléments des politiques IAM : variables et balises dans le Guide de l'utilisateur IAM.

AWS prend en charge les clés de condition globales et les clés de condition spécifiques à un service. Pour afficher toutes les clés de condition globales AWS, consultez Clés de contexte de condition globale AWS dans le Guide de l'utilisateur IAM.

Pour obtenir la liste des clés de GameLift condition Amazon, consultez la section Clés de condition pour Amazon GameLift dans la référence d'autorisation de service. Pour savoir avec quelles actions et ressources vous pouvez utiliser une clé de condition, consultez Actions définies par Amazon GameLift.

Pour consulter des exemples de politiques GameLift basées sur l'identité d'Amazon, consultez. Exemples de politiques basées sur l'identité pour Amazon GameLift

ACL sur Amazon GameLift

Prend en charge les listes ACL	Non

Les listes de contrôle d'accès (ACL) vérifient quels principaux (membres de compte, utilisateurs ou rôles) ont l'autorisation d'accéder à une ressource. Les listes de contrôle d'accès sont similaires aux

politiques basées sur les ressources, bien qu'elles n'utilisent pas le format de document de politique JSON.

ABAC avec Amazon GameLift

Prend en charge ABAC (étiquettes dans les politiques)	Oui

Le contrôle d'accès par attributs (ABAC) est une stratégie d'autorisation qui définit des autorisations en fonction des attributs. Dans AWS, ces attributs sont appelés étiquettes. Vous pouvez attacher des étiquettes à des entités IAM (utilisateurs ou rôles), ainsi qu'à de nombreuses ressources AWS. L'étiquetage des entités et des ressources est la première étape d'ABAC. Vous concevez ensuite des politiques ABAC pour autoriser des opérations quand l'identification du principal correspond à celle de la ressource à laquelle il tente d'accéder.

L'ABAC est utile dans les environnements qui connaissent une croissance rapide et pour les cas où la gestion des politiques devient fastidieuse.

Pour contrôler l'accès basé sur des balises, vous devez fournir les informations de balise dans l'élément de condition d'une politique utilisant les clés de condition `aws:ResourceTag/key-name`, `aws:RequestTag/key-name` ou `aws:TagKeys`.

Si un service prend en charge les trois clés de condition pour tous les types de ressources, alors la valeur pour ce service est Oui. Si un service prend en charge les trois clés de condition pour certains types de ressources uniquement, la valeur est Partielle.

Pour plus d'informations sur l'ABAC, consultez Qu'est-ce que le contrôle d'accès basé sur les attributs (ABAC) ? dans le Guide de l'utilisateur IAM. Pour accéder à un didacticiel décrivant les étapes de configuration de l'ABAC, consultez Utilisation du contrôle d'accès par attributs (ABAC) dans le Guide de l'utilisateur IAM.

Pour un exemple de politique basée sur l'identité qui limite l'accès à une ressource en fonction des balises associées à cette ressource, voir. Afficher les GameLift flottes Amazon en fonction des tags

Utilisation d'informations d'identification temporaires avec Amazon GameLift

Prend en charge les informations d'identification temporaires	Oui

Certains Services AWS ne fonctionnent pas quand vous vous connectez à l'aide d'informations d'identification temporaires. Pour plus d'informations, notamment sur les Services AWS qui fonctionnent avec des informations d'identification temporaires, consultez Services AWS qui fonctionnent avec IAM dans le Guide de l'utilisateur IAM.

Vous utilisez des informations d'identification temporaires quand vous vous connectez à la AWS Management Console en utilisant toute méthode autre qu'un nom d'utilisateur et un mot de passe. Par exemple, lorsque vous accédez à AWS en utilisant le lien d'authentification unique (SSO) de votre société, ce processus crée automatiquement des informations d'identification temporaires. Vous créez également automatiquement des informations d'identification temporaires lorsque vous vous connectez à la console en tant qu'utilisateur, puis changez de rôle. Pour plus d'informations sur le changement de rôle, consultez Changement de rôle (console) dans le Guide de l'utilisateur IAM.

Vous pouvez créer manuellement des informations d'identification temporaires à l'aide d'AWS CLI ou de l'API AWS. Vous pouvez ensuite utiliser ces informations d'identification temporaires pour accéder à AWS. AWS recommande de générer des informations d'identification temporaires de façon dynamique au lieu d'utiliser des clés d'accès à long terme. Pour plus d'informations, consultez Informations d'identification de sécurité temporaires dans IAM.

Autorisations principales interservices pour Amazon GameLift

Prend en charge les sessions d'accès direct (FAS)	Oui

Lorsque vous vous servez d'un utilisateur IAM ou d'un rôle IAM pour accomplir des actions dans AWS, vous êtes considéré comme un principal. Lorsque vous utilisez certains services, l'action que vous effectuez est susceptible de lancer une autre action dans un autre service. FAS utilise les autorisations du principal appelant Service AWS, combinées à la demande Service AWS pour effectuer des demandes aux services en aval. Les demandes FAS ne sont formulées que lorsqu'un service reçoit une demande qui, pour aboutir, a besoin d'interagir avec d'autres ressources ou Services AWS. Dans ce cas, vous devez disposer d'autorisations nécessaires pour effectuer les deux actions. Pour plus de détails sur une politique lors de la formulation de demandes FAS, consultez Transmission des sessions d'accès.

Rôles de service pour Amazon GameLift

Prend en charge les fonctions du service	Oui

Un rôle de service est un rôle IAM qu'un service endosse pour accomplir des actions en votre nom. Un administrateur IAM peut créer, modifier et supprimer une fonction du service à partir d'IAM. Pour plus d'informations, consultez Création d'un rôle pour la délégation d'autorisations à un Service AWS dans le Guide de l'utilisateur IAM.

> **Warning**
>
> La modification des autorisations associées à un rôle de service peut perturber les GameLift fonctionnalités d'Amazon. Modifiez les rôles de service uniquement lorsque Amazon GameLift fournit des conseils à cet effet.

Autorisez vos serveurs de jeu GameLift hébergés par Amazon à accéder à d'autres AWS ressources, telles qu'une AWS Lambda fonction ou une base de données Amazon DynamoDB. Les serveurs de jeu étant hébergés sur des flottes gérées par Amazon GameLift , vous avez besoin d'un rôle de service qui donne à Amazon un accès GameLift limité à vos autres AWS ressources. Pour plus d'informations, consultez Communiquez avec les autres AWS ressources de vos flottes.

Rôles liés à un service pour Amazon GameLift

Prend en charge les rôles liés à un service.	Non

Un rôle lié à un service est un type de rôle de service lié à un Service AWS. Le service peut endosser le rôle afin d'effectuer une action en votre nom. Les rôles liés à un service s'affichent dans votre Compte AWS et sont détenus par le service. Un administrateur IAM peut consulter, mais ne peut pas modifier, les autorisations concernant les rôles liés à un service.

Pour plus de détails sur la création ou la gestion des rôles liés à un service, consultez la section AWSServices compatibles avec IAM dans le Guide de l'utilisateur d'IAM. Recherchez un service dans le tableau qui inclut un `Yes` dans la colonne Rôles liés au service. Choisissez Oui pour consulter la documentation relative aux rôles liés à un service pour ce service.

Exemples de politiques basées sur l'identité pour Amazon GameLift

Par défaut, les utilisateurs et les rôles ne sont pas autorisés à créer ou à modifier GameLift des ressources Amazon. Ils ne peuvent pas non plus exécuter des tâches à l'aide de la AWS Management Console, de l'AWS Command Line Interface (AWS CLI) ou de l'API AWS. Pour octroyer aux utilisateurs des autorisations d'effectuer des actions sur les ressources dont ils ont besoin, un

administrateur IAM doit créer des politiques IAM. L'administrateur peut ensuite ajouter les politiques IAM aux rôles et les utilisateurs peuvent endosser les rôles.

Pour apprendre à créer une politique basée sur l'identité IAM à l'aide de ces exemples de documents de politique JSON, consultez Création de politiques IAM dans le Guide de l'utilisateur IAM.

Pour plus de détails sur les actions et les types de ressources définis par Amazon GameLift, y compris le format des ARN pour chacun des types de ressources, consultez la section Actions, ressources et clés de condition pour Amazon GameLift dans le Service Authorization Reference.

Rubriques

Bonnes pratiques en matière de politiques

Les politiques basées sur l'identité déterminent si quelqu'un peut créer, accéder ou supprimer GameLift des ressources Amazon dans votre compte. Ces actions peuvent entraîner des frais pour votre Compte AWS. Lorsque vous créez ou modifiez des politiques basées sur l'identité, suivez ces instructions et recommandations :

- Démarrer avec AWS gérées et évoluez vers les autorisations de moindre privilège - Pour commencer à accorder des autorisations à vos utilisateurs et charges de travail, utilisez les politiques gérées AWS qui accordent des autorisations dans de nombreux cas d'utilisation courants. Elles sont disponibles dans votre Compte AWS. Nous vous recommandons de réduire encore les autorisations en définissant des politiques gérées par le client AWS qui sont spécifiques à vos cas d'utilisation. Pour de plus amples informations, consultez AWS Politiques gérées ou AWS Politiques gérées pour les activités professionnelles dans le Guide de l'utilisateur IAM.
- Accorder les autorisations de moindre privilège - Lorsque vous définissez des autorisations avec des politiques IAM, accordez uniquement les autorisations nécessaires à l'exécution d'une seule tâche. Pour ce faire, vous définissez les actions qui peuvent être entreprises sur des ressources

spécifiques dans des conditions spécifiques, également appelées autorisations de moindre privilège. Pour plus d'informations sur l'utilisation d'IAM pour appliquer des autorisations, consultez Politiques et autorisations dans IAM dans le Guide de l'utilisateur IAM.

- Utiliser des conditions dans les politiques IAM pour restreindre davantage l'accès - Vous pouvez ajouter une condition à vos politiques afin de limiter l'accès aux actions et aux ressources. Par exemple, vous pouvez écrire une condition de politique pour spécifier que toutes les demandes doivent être envoyées via SSL. Vous pouvez également utiliser des conditions pour accorder l'accès aux actions de service si elles sont utilisées via un Service AWS spécifique, comme AWS CloudFormation. Pour plus d'informations, consultez Conditions pour éléments de politique JSON IAM dans le Guide de l'utilisateur IAM.
- Utilisez IAM Access Analyzer pour valider vos politiques IAM afin de garantir des autorisations sécurisées et fonctionnelles - IAM Access Analyzer valide les politiques nouvelles et existantes de manière à ce que les politiques IAM respectent le langage de politique IAM (JSON) et les bonnes pratiques IAM. IAM Access Analyzer fournit plus de 100 vérifications de politiques et des recommandations exploitables pour vous aider à créer des politiques sécurisées et fonctionnelles. Pour plus d'informations, consultez Validation de politique IAM Access Analyzer dans le Guide de l'utilisateur IAM.
- Authentification multifactorielle (MFA) nécessaire : si vous avez un scénario qui nécessite des utilisateurs IAM ou un utilisateur root dans votre Compte AWS, activez l'authentification multifactorielle pour une sécurité renforcée. Pour exiger le MFA lorsque des opérations d'API sont appelées, ajoutez des conditions MFA à vos politiques. Pour plus d'informations, consultez Configuration de l'accès aux API protégé par MFA dans le Guide de l'utilisateur IAM.

Pour plus d'informations sur les bonnes pratiques dans IAM, consultez Bonnes pratiques de sécurité dans IAM dans le Guide de l'utilisateur IAM.

Utilisation de la GameLift console Amazon

Pour accéder à la GameLift console Amazon, vous devez disposer d'un ensemble minimal d'autorisations. Ces autorisations doivent vous permettre de répertorier et de consulter les informations relatives aux GameLift ressources Amazon présentes dans votreCompte AWS. Si vous créez une stratégie basée sur l'identité qui est plus restrictive que l'ensemble minimum d'autorisations requis, la console ne fonctionnera pas comme prévu pour les entités (utilisateurs ou rôles) tributaires de cette stratégie.

Pour garantir que ces entités peuvent toujours utiliser la GameLift console Amazon, ajoutez des autorisations aux utilisateurs et aux groupes en utilisant la syntaxe décrite dans les exemples

suivants et dansExemples d'autorisations d'administrateur. Pour plus d'informations, consultez Gérer les autorisations des utilisateurs pour Amazon GameLift.

Les utilisateurs qui travaillent avec Amazon GameLift via des opérations AWS CLI d'AWSAPI n'ont pas besoin d'autorisations de console minimales. Au lieu de cela, vous pouvez limiter l'accès aux seules opérations que l'utilisateur doit effectuer. Par exemple, un utilisateur joueur, agissant pour le compte de clients du jeu, a besoin d'un accès pour demander des sessions de jeu, placer des joueurs dans des jeux et effectuer d'autres tâches.

Pour plus d'informations sur les autorisations requises pour utiliser toutes les fonctionnalités de GameLift la console Amazon, consultez la syntaxe des autorisations pour les administrateurs dansExemples d'autorisations d'administrateur.

Autorisation accordée aux utilisateurs pour afficher leurs propres autorisations

Cet exemple montre comment créer une politique qui permet aux utilisateurs IAM d'afficher les politiques en ligne et gérées attachées à leur identité d'utilisateur. Cette politique inclut les autorisations nécessaires pour réaliser cette action sur la console ou par programmation à l'aide de l'AWS CLI ou de l'API 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": "*"
        }
    ]
}
```

Autoriser l'accès de joueurs pour les sessions de jeu

Pour placer des joueurs dans des sessions de jeu, les clients du jeu et les services principaux ont besoin d'autorisations. Pour des exemples de politiques pour ces scénarios, voirExemples d'autorisations utilisateur pour les joueurs.

Autoriser l'accès à une GameLift file d'attente Amazon

L'exemple suivant fournit à un utilisateur l'accès à une file d' GameLiftattente Amazon spécifique.

Cette politique accorde à l'utilisateur l'autorisation d'ajouter, de mettre à jour et de supprimer des destinations de file d'attente avec les actions suivantes : `gamelift:UpdateGameSessionQueue``gamelift:DeleteGameSessionQueue`, et`gamelift:DescribeGameSessionQueues`. Comme indiqué, cette politique utilise l'`Resource`élément pour limiter l'accès à une seule file d'attente :`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"
      }
   ]
}
```

Afficher les GameLift flottes Amazon en fonction des tags

Vous pouvez utiliser des conditions dans votre politique basée sur l'identité pour contrôler l'accès aux GameLift ressources Amazon en fonction de balises. Cet exemple montre comment créer une politique qui permet de consulter une flotte si le `Owner` tag correspond au nom d'utilisateur de l'utilisateur. Cette politique accorde également les autorisations nécessaires pour effectuer cette opération dans la console.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListFleetsInConsole",
            "Effect": "Allow",
            "Action": "gamelift:ListFleets",
            "Resource": "*"
        },
        {
            "Sid": "ViewFleetIfOwner",
            "Effect": "Allow",
            "Action": "gamelift:DescribeFleetAttributes",
            "Resource": "arn:aws:gamelift:*:*:fleet/*",
            "Condition": {
                "StringEquals": {"gamelift:ResourceTag/Owner": "${aws:username}"}
            }
        }
    ]
}
```

Accédez à un fichier de compilation de jeu dans Amazon S3

Après avoir intégré votre serveur de jeu à Amazon GameLift, téléchargez les fichiers de compilation sur Amazon S3. Pour GameLift qu'Amazon puisse accéder aux fichiers de compilation, appliquez la politique suivante.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": "arn:aws:s3:::bucket-name/object-name"
        }
    ]
}
```

Pour plus d'informations sur le téléchargement de fichiers de GameLift jeux Amazon, consultezTéléchargez une version de serveur personnalisée sur Amazon GameLift.

Résolution des problèmes d' GameLift identité et d'accès à Amazon

Utilisez les informations suivantes pour vous aider à diagnostiquer et à résoudre les problèmes courants que vous pouvez rencontrer lorsque vous travaillez avec Amazon GameLift et AWS Identity and Access Management (IAM).

Rubriques

- Je ne suis pas autorisé à effectuer une action sur Amazon GameLift
- Je ne suis pas autorisé à effectuer iam : PassRole
- Je souhaite autoriser des personnes extérieures Compte AWS à moi à accéder à mes GameLift ressources Amazon

Je ne suis pas autorisé à effectuer une action sur Amazon GameLift

S'il vous AWS Management Console indique que vous n'êtes pas autorisé à effectuer une action, contactez l'administrateur de votre AWS compte pour obtenir de l'aide. Votre administrateur est la personne qui vous a fourni vos informations de connexion.

L'exemple d'erreur suivant se produit lorsque l'utilisateur `mateojackson` IAM essaie d'utiliser la console pour afficher les détails d'une file d'attente mais ne dispose pas des `gamelift:DescribeGameSessionQueues` autorisations nécessaires :

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 gamelift:DescribeGameSessionQueues on resource: examplequeue123
```

Dans ce cas, Mateo demande à son administrateur de mettre à jour ses politiques pour lui permettre d'accéder en lecture à la `examplequeue123` ressource utilisant l'`gamelift:DescribeGameSessionQueues`action.

Je ne suis pas autorisé à effectuer iam : PassRole

Si vous recevez un message d'erreur indiquant que vous n'êtes pas autorisé à effectuer l'`iam:PassRole`action, vos politiques doivent être mises à jour pour vous permettre de transmettre un rôle à Amazon GameLift.

Certains Services AWS vous permettent de transmettre un rôle existant à ce service, au lieu de créer une nouvelle fonction du service ou rôle lié à un service. Pour ce faire, un utilisateur doit disposer des autorisations nécessaires pour transmettre le rôle au service.

L'exemple d'erreur suivant se produit lorsqu'un utilisateur IAM nommé `marymajor` essaie d'utiliser la console pour effectuer une action sur Amazon GameLift. Toutefois, l’action nécessite que le service ait des autorisations accordées par une fonction du service. Mary ne dispose pas des autorisations nécessaires pour transférer le rôle au service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

Dans ce cas, les politiques de Mary doivent être mises à jour pour lui permettre d’exécuter l’action `iam:PassRole`.

Si vous avez encore besoin d’aide, contactez votre administrateur AWS. Votre administrateur vous a fourni vos informations de connexion.

Je souhaite autoriser des personnes extérieures Compte AWS à moi à accéder à mes GameLift ressources Amazon

Vous pouvez créer un rôle que les utilisateurs provenant d'autres comptes ou les personnes extérieures à votre organisation pourront utiliser pour accéder à vos ressources. Vous pouvez spécifier qui est autorisé à assumer le rôle. Pour les services qui prennent en charge les politiques basées sur les ressources ou les listes de contrôle d'accès (ACL), vous pouvez utiliser ces politiques pour donner l'accès à vos ressources.

Pour en savoir plus, consultez les éléments suivants :

- Pour savoir si Amazon GameLift prend en charge ces fonctionnalités, consultezComment Amazon GameLift travaille avec IAM.
- Pour savoir comment octroyer l'accès à vos ressources à des Comptes AWS dont vous êtes propriétaire, consultez la section Fournir l'accès à un utilisateur IAM dans un autre Compte AWS que vous possédez dans le Guide de l'utilisateur IAM.
- Pour savoir comment octroyer l'accès à vos ressources à des tiers Comptes AWS, consultez Fournir l'accès aux Comptes AWS appartenant à des tiers dans le Guide de l'utilisateur IAM.
- Pour savoir comment fournir un accès par le biais de la fédération d'identité, consultez Fournir un accès à des utilisateurs authentifiés en externe (fédération d'identité) dans le Guide de l'utilisateur IAM.
- Pour découvrir quelle est la différence entre l'utilisation des rôles et l'utilisation des politiques basées sur les ressources pour l'accès entre comptes, consultez Différence entre les rôles IAM et les politiques basées sur les ressources dans le Guide de l'utilisateur IAM.

Enregistrement et surveillance avec Amazon GameLift

La surveillance joue un rôle important dans le maintien de la fiabilité, de la disponibilité et des performances d'Amazon GameLift et de vos AWS solutions. Vous devez recueillir les données de surveillance de toutes les parties de votre solution AWS de telle sorte que vous puissiez déboguer plus facilement un éventuelle défaillance multipoint.

AWSet Amazon GameLift proposent plusieurs outils pour surveiller les ressources d'hébergement de vos jeux et répondre aux incidents potentiels.

CloudWatchAlarmes Amazon

À l'aide des CloudWatch alarmes Amazon, vous surveillez une seule métrique sur une période que vous spécifiez. Si la métrique dépasse un seuil donné, une notification est envoyée à une rubrique Amazon SNS ou à une stratégie AWS Auto Scaling. CloudWatchles alarmes sont déclenchées lorsque leur état change et sont maintenues pendant un certain nombre de périodes, et non parce qu'elles se trouvent dans un état particulier. Pour plus d'informations, veuillez consulter Surveillez Amazon GameLift avec Amazon CloudWatch.

AWSCloudTrailJournaux

CloudTrailfournit un enregistrement des actions effectuées par un utilisateur, un rôle ou un AWS service sur AmazonGameLift. À l'aide des informations collectées parCloudTrail, vous pouvez déterminer la demande qui a été faite à AmazonGameLift, l'adresse IP à partir de laquelle la demande a été faite, qui l'a faite, quand elle a été faite et des détails supplémentaires. Pour plus d'informations, veuillez consulter Enregistrement des appels GameLift d'API Amazon avec AWS CloudTrail.

Validation de conformité pour Amazon GameLift

Amazon n' GameLift est concerné par aucun programme de AWS conformité.

Pour savoir si un programme Services AWS de conformité Service AWS s'inscrit dans le champ d'application de programmes de conformité spécifiques, consultez Services AWS la section de conformité et sélectionnez le programme de conformité qui vous intéresse. Pour des informations générales, voir Programmes de AWS conformité Programmes AWS de .

Vous pouvez télécharger des rapports d'audit tiers à l'aide de AWS Artifact. Pour plus d'informations, voir Téléchargement de rapports dans AWS Artifact .

Votre responsabilité en matière de conformité lors de l'utilisation Services AWS est déterminée par la sensibilité de vos données, les objectifs de conformité de votre entreprise et les lois et réglementations applicables. AWS fournit les ressources suivantes pour faciliter la mise en conformité :

- Guides de démarrage rapide sur la sécurité et la conformité : ces guides de déploiement abordent les considérations architecturales et indiquent les étapes à suivre pour déployer des environnements de base axés sur AWS la sécurité et la conformité.
- Architecture axée sur la sécurité et la conformité HIPAA sur Amazon Web Services : ce livre blanc décrit comment les entreprises peuvent créer des applications AWS conformes à la loi HIPAA.

> **Note**
>
> Tous ne Services AWS sont pas éligibles à la loi HIPAA. Pour plus d'informations, consultez le HIPAA Eligible Services Reference.

- AWS Ressources de https://aws.amazon.com/compliance/resources/ de conformité — Cette collection de classeurs et de guides peut s'appliquer à votre secteur d'activité et à votre région.
- AWS Guides de conformité destinés aux clients — Comprenez le modèle de responsabilité partagée sous l'angle de la conformité. Les guides résument les meilleures pratiques en matière de sécurisation Services AWS et décrivent les directives relatives aux contrôles de sécurité dans de nombreux cadres (notamment le National Institute of Standards and Technology (NIST), le Payment Card Industry Security Standards Council (PCI) et l'Organisation internationale de normalisation (ISO)).
- Évaluation des ressources à l'aide des règles du guide du AWS Config développeur : le AWS Config service évalue dans quelle mesure les configurations de vos ressources sont conformes aux pratiques internes, aux directives du secteur et aux réglementations.
- AWS Security Hub— Cela Service AWS fournit une vue complète de votre état de sécurité interne AWS. Security Hub utilise des contrôles de sécurité pour évaluer vos ressources AWS et vérifier votre conformité par rapport aux normes et aux bonnes pratiques du secteur de la sécurité. Pour obtenir la liste des services et des contrôles pris en charge, consultez Référence des contrôles Security Hub.
- Amazon GuardDuty — Cela Service AWS détecte les menaces potentielles qui pèsent sur vos charges de travail Comptes AWS, vos conteneurs et vos données en surveillant votre environnement pour détecter toute activité suspecte et malveillante. GuardDuty peut vous aider à répondre à diverses exigences de conformité, telles que la norme PCI DSS, en répondant aux exigences de détection des intrusions imposées par certains cadres de conformité.
- AWS Audit Manager— Cela vous Service AWS permet d'auditer en permanence votre AWS utilisation afin de simplifier la gestion des risques et la conformité aux réglementations et aux normes du secteur.

Résilience chez Amazon GameLift

Si vous utilisez Amazon GameLift FleetIQ en tant que fonctionnalité autonome avec Amazon EC2, consultez la section Sécurité dans Amazon EC2 dans le guide de l'utilisateur d'Amazon EC2.

L'infrastructure AWS mondiale est construite autour des AWS régions et des zones de disponibilité. AWS Les régions fournissent plusieurs zones de disponibilité physiquement séparées et isolées, connectées par un réseau à faible latence, à haut débit et hautement redondant. Avec les zones de disponibilité, vous pouvez concevoir et exploiter des applications et des bases de données qui basculent automatiquement d'une zone à l'autre sans interruption. Les zones de disponibilité sont davantage disponibles, tolérantes aux pannes et ont une plus grande capacité de mise à l'échelle que les infrastructures traditionnelles à un ou plusieurs centres de données.

Pour plus d'informations sur AWS les régions et les zones de disponibilité, consultez la section Infrastructure AWS globale.

Outre l'infrastructure AWS mondiale, Amazon GameLift propose les fonctionnalités suivantes pour répondre à vos besoins en matière de résilience des données :

- Files d'attente multirégionales : les files d'attente des sessions de GameLift jeu Amazon sont utilisées pour créer de nouvelles sessions de jeu avec les ressources d'hébergement disponibles. Les files d'attente couvrant plusieurs régions peuvent rediriger les placements de session de jeu en cas de panne au niveau d'une région. Pour de plus amples informations et connaître les bonnes pratiques relatives à la création de files d'attente de session de jeu, veuillez consulter Conception d'une file d'attente de sessions de jeu.
- Dimensionnement automatique de la capacité : maintenez l'intégrité et la disponibilité de vos ressources d'hébergement à l'aide des outils de GameLift dimensionnement d'Amazon. Ces outils offrent une palette d'options qui vous permettent d'ajuster la capacité de votre parc en fonction des besoins de votre jeu et des joueurs. Pour plus d'informations sur le dimensionnement, consultez Élargir la capacité GameLift d'hébergement d'Amazon.
- Distribution entre instances : Amazon GameLift répartit le trafic entrant sur plusieurs instances, en fonction de la taille du parc. Pour les jeux en production, une bonne pratique consiste à disposer de plusieurs instances pour conserver la disponibilité au cas où une instance devient non saine ou ne répond pas.
- Stockage Amazon S3 : les versions de serveurs de jeux et les scripts téléchargés sur Amazon GameLift sont stockés dans Amazon S3 à l'aide de la classe de stockage Standard, qui utilise plusieurs réplications de centres de données pour améliorer la résilience. Les journaux des sessions de jeu sont également stockés dans Amazon S3 à l'aide de la classe de stockage Standard.

Sécurité de l'infrastructure sur Amazon GameLift

Si vous utilisez Amazon GameLift FleetIQ en tant que fonctionnalité autonome avec Amazon EC2, consultez la section Sécurité dans Amazon EC2 dans le guide de l'utilisateur d'Amazon EC2.

En tant que service géré, Amazon GameLift est protégé par les procédures de sécurité du réseau AWS mondial décrites dans le livre blanc Amazon Web Services : présentation des processus de sécurité.

Vous utilisez des appels d'API AWS publiés pour accéder à Amazon GameLift via le réseau. Les clients doivent prendre en charge le protocole TLS (Transport Layer Security) 1.2 ou version ultérieure. Nous recommandons TLS 1.3 ou version ultérieure. Les clients doivent aussi prendre en charge les suites de chiffrement PFS (Perfect Forward Secrecy) comme Ephemeral Diffie-Hellman (DHE) ou Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). La plupart des systèmes modernes tels que Java 7 et les versions ultérieures prennent en charge ces modes.

En outre, les demandes doivent être signées à l'aide d'un ID de clé d'accès et d'une clé d'accès secrète associée à un principal IAM. Vous pouvez également utiliser AWS Security Token Service (AWS STS) pour générer des informations d'identification de sécurité temporaires et signer les demandes.

Le GameLift service Amazon place toutes les flottes dans des clouds privés virtuels (VPC) Amazon afin que chaque flotte existe dans une zone logiquement isolée du cloud. AWS Vous pouvez utiliser GameLift les politiques d'Amazon pour contrôler l'accès depuis des points de terminaison VPC spécifiques ou des VPC spécifiques. En fait, cela isole l'accès réseau à une GameLift ressource Amazon donnée uniquement du VPC spécifique au sein AWS du réseau. Lorsque vous créez un parc, vous spécifiez une plage de numéros de port et d'adresses IP. Ces plages limitent la façon dont le trafic entrant peut accéder aux serveurs de jeux hébergés sur un VPC de parc. Utilisez les bonnes pratiques en matière de sécurité standard lorsque vous choisissez les paramètres d'accès au parc.

Analyse de configuration et de vulnérabilité sur Amazon GameLift

Si vous utilisez Amazon GameLift FleetIQ en tant que fonctionnalité autonome avec Amazon EC2, consultez la section Sécurité dans Amazon EC2 dans le guide de l'utilisateur d'Amazon EC2.

La configuration et les contrôles informatiques sont une responsabilité partagée entre AWS et vous, notre client. Pour plus d'informations, consultez le modèle de responsabilité AWS partagée. AWS gère les tâches de sécurité de base telles que l'application de correctifs au système d'exploitation client (OS) et aux bases de données, la configuration du pare-feu et la reprise après sinistre. Ces

procédures ont été vérifiées et certifiées par les tiers appropriés. Pour plus de détails, consultez la ressource suivante : Amazon Web Services : présentation des processus de sécurité (livre blanc).

Les bonnes pratiques de sécurité suivantes concernent également la configuration et l'analyse des vulnérabilités sur Amazon GameLift :

- Les clients sont responsables de la gestion des logiciels déployés sur les GameLift instances Amazon pour l'hébergement de jeux. En particulier :
  - Les logiciels d'application de serveur de jeux fournis par le client doivent être gérés, y compris les mises à jour et les correctifs de sécurité. Pour mettre à jour le logiciel du serveur de jeu, téléchargez une nouvelle version sur Amazon GameLift, créez une nouvelle flotte et redirigez le trafic vers cette nouvelle flotte.
  - L'AMI (Amazon Machine Image) de base, qui inclut le système d'exploitation, est mise à jour uniquement lorsqu'un nouveau parc est créé. Pour appliquer des correctifs, mettre à jour et sécuriser le système d'exploitation et les autres applications faisant partie de l'AMI, recyclez régulièrement les parcs, indépendamment des mises à jour du serveur de jeu.
- Les clients devraient envisager de mettre régulièrement à jour leurs jeux avec les dernières versions du SDK, notamment le AWS SDK, le SDK Amazon GameLift Server et le SDK Amazon GameLift Client pour serveurs en temps réel.

Bonnes pratiques de sécurité pour Amazon GameLift

Si vous utilisez Amazon GameLift FleetIQ en tant que fonctionnalité autonome avec Amazon EC2, consultez la section Sécurité dans Amazon EC2 dans le guide de l'utilisateur d'Amazon EC2.

Amazon GameLift propose un certain nombre de fonctionnalités de sécurité à prendre en compte lors de l'élaboration et de la mise en œuvre de vos propres politiques de sécurité. Les bonnes pratiques suivantes doivent être considérées comme des instructions générales et ne représentent pas une solution de sécurité complète. Étant donné que ces bonnes pratiques peuvent ne pas être appropriées ou suffisantes pour votre environnement, considérez-les comme des remarques utiles plutôt que comme des recommandations.

N'ouvrez pas de ports vers Internet

Nous vous déconseillons vivement d'ouvrir des ports vers Internet car cela représente un risque pour la sécurité. Par exemple, si vous avez l'habitude UpdateFleetPortSettingsd'ouvrir un port de poste de travail distant comme celui-ci :

```
{
  "FleetId": "<fleet identifier>",
  "InboundPermissionAuthorizations": [
      {
        "FromPort": 3389,
        "IpRange": "0.0.0.0/0",
        "Protocol": "RDP",
        "ToPort": 3389
      }
  ]
}
```

alors vous autorisez n'importe qui sur Internet à accéder à l'instance.

Ouvrez plutôt le port avec une adresse IP ou une plage d'adresses spécifiques. Par exemple, comme ceci :

```
{
  "FleetId": "<fleet identifier>",
  "InboundPermissionAuthorizations": [
      {
        "FromPort": 3389,
        "IpRange": "54.186.139.221/32",
        "Protocol": "TCP",
        "ToPort": 3389
      }
  ]
}
```

En savoir plus

Pour plus d'informations sur la manière de sécuriser votre utilisation GameLift d'Amazon, consultez le pilier AWS Well-Architected Tool Sécurité. .

Guides GameLift de référence Amazon

Cette section contient de la documentation de référence pour l'utilisation d'AmazonGameLift.

Rubriques

- Référence GameLift de l'API du service Amazon (AWSSDK)
- Référence des serveurs Amazon GameLift Realtime
- Référence GameLift du SDK pour serveurs Amazon
- Événements de placement de sessions de jeu

Référence GameLift de l'API du service Amazon (AWSSDK)

Cette rubrique fournit une liste des opérations d'API basées sur les tâches à utiliser avec les solutions d'hébergement GameLift géré Amazon, y compris l'hébergement de serveurs de jeux personnalisés et de serveurs en temps réel. Ces opérations sont regroupées dans le AWS SDK dans l'espace de `aws.gamelift` nommage. Téléchargez le AWS SDK ou consultez la documentation de référence de GameLift l'API Amazon.

L'API inclut deux ensembles d'opérations pour l'hébergement de jeux gérés :

- Configuration et gestion des ressources GameLift d'hébergement Amazon
- Démarrez des sessions de jeu et rejoignez des joueurs

L'API Amazon GameLift Service contient également des opérations à utiliser avec d'autres GameLift outils et solutions Amazon. Pour obtenir la liste des API FleetIQ, consultez la section Actions de l'API FleetIQ. Pour une liste des FlexMatch API pour le matchmaking, voir Actions de FlexMatch l'API.

Configuration et gestion des ressources GameLift d'hébergement Amazon

Faites appel à ces opérations pour configurer les ressources d'hébergement de vos serveurs de jeu, adapter la capacité à la demande des joueurs, accéder aux indicateurs de performance et d'utilisation, etc. Ces opérations d'API sont utilisées avec les serveurs de jeux hébergés sur AmazonGameLift, y compris les serveurs en temps réel. Vous pouvez utiliser la GameLiftconsole Amazon pour la plupart des tâches de gestion des ressources, ou vous pouvez appeler le service à l'aide de l'outil AWS Command Line Interface (AWS CLI) ou du AWS SDK.

Préparer les serveurs de jeu pour le déploiement

Téléchargez et configurez le code du serveur de jeu de votre jeu en vue du déploiement et du lancement sur les ressources d'hébergement.

Gérez les versions personnalisées de serveurs de jeu

- upload-build : chargez des fichiers de build depuis un chemin local et créez une nouvelle ressource de GameLift build Amazon. Cette opération, disponible uniquement sous forme de AWS CLI commande, est la méthode la plus courante pour télécharger des versions de serveurs de jeu.
- CreateBuild— Créez une nouvelle version à l'aide de fichiers stockés dans un compartiment Amazon S3.
- ListBuilds— Obtenez une liste de toutes les versions téléchargées dans une GameLift région Amazon.
- DescribeBuild— Récupère les informations associées à une version.
- UpdateBuild— Modifiez les métadonnées de build, y compris le nom et la version du build.
- DeleteBuild— Supprime une version d'AmazonGameLift.

Gestion des scripts de configuration des serveurs en temps réel

- CreateScript— Téléchargez JavaScript des fichiers et créez une nouvelle ressource de GameLift script Amazon.
- ListScripts— Obtenez la liste de tous les scripts en temps réel chargés dans une GameLift région Amazon.
- DescribeScript— Récupère les informations associées à un script en temps réel.
- UpdateScript— Modifiez les métadonnées du script et chargez le contenu du script révisé.
- DeleteScript— Supprime un script en temps réel d'AmazonGameLift.

Configuration des ressources informatiques pour l'hébergement

Configurez les ressources d'hébergement et déployez-les à l'aide de la version de votre serveur de jeu ou du script de configuration en temps réel.

Création et gestion de flottes

- CreateFleet— Configurez et déployez un nouveau GameLift parc de ressources informatiques Amazon pour faire fonctionner vos serveurs de jeux. Une fois déployés, les serveurs de jeu sont automatiquement lancés tels que configurés et prêts à héberger des sessions de jeu.
- ListFleets— Obtenez la liste de toutes les flottes d'une GameLift région amazonienne.
- DeleteFleet— Mettez fin à une flotte qui ne gère plus de serveurs de jeu ou n'héberge plus de joueurs.
- Afficher/mettre à jour l'emplacement de la flotte.
  - CreateFleetLocations— Ajoutez des sites distants à une flotte existante qui prend en charge plusieurs sites
  - DescribeFleetLocationAttributes— Obtenez la liste de tous les sites distants d'une flotte et consultez l'état actuel de chaque site.
  - DeleteFleetLocations— Supprimez les sites distants d'un parc qui prend en charge plusieurs sites.
- Afficher/mettre à jour les configurations de flotte.
  - DescribeFleetAttributes/UpdateFleetAttributes— Affichez ou modifiez les métadonnées et les paramètres d'une flotte pour la protection des sessions de jeu et les limites de création de ressources.
  - DescribeFleetPortSettings/UpdateFleetPortSettings— Affichez ou modifiez les autorisations entrantes (adresses IP et plages de paramètres de port) autorisées pour une flotte.
  - DescribeRuntimeConfiguration/UpdateRuntimeConfiguration— Affichez ou modifiez les processus du serveur (et leur nombre) à exécuter sur chaque instance d'un parc.

Gestion de la capacité de la flotte

- DescribeEC2 InstanceLimits — Récupère le nombre maximum d'instances autorisées pour le AWS compte actuel et le niveau d'utilisation actuel.
- DescribeFleetCapacity— Récupère les paramètres de capacité actuels pour la région d'origine d'une flotte.
- DescribeFleetLocationCapacity— Récupérez les paramètres de capacité actuels pour chaque site d'une flotte multi-sites.
- UpdateFleetCapacity— Ajustez manuellement les paramètres de capacité d'une flotte.
- Configurez la mise à l'échelle automatique :

- PutScalingPolicy— Activez le dimensionnement automatique basé sur les cibles, créez une politique de dimensionnement automatique personnalisée ou mettez à jour une politique existante.
- DescribeScalingPolicies— Récupère une politique de mise à l'échelle automatique existante.
- DeleteScalingPolicy— Supprimez une politique de mise à l'échelle automatique et empêchez-la d'affecter la capacité d'une flotte.
- StartFleetActions— Redémarrez les politiques de mise à l'échelle automatique d'une flotte.
- StopFleetActions— Suspendez les politiques de mise à l'échelle automatique d'une flotte.

Surveillance de l'activité de la flotte.

- DescribeFleetUtilization— Récupérez des statistiques sur le nombre de processus du serveur, de sessions de jeu et de joueurs actuellement actifs sur une flotte.
- DescribeFleetLocationUtilization— Récupérez les statistiques d'utilisation pour chaque site d'une flotte multi-sites.
- DescribeFleetEvents— Afficher les événements enregistrés pour une flotte au cours d'une période spécifiée.
- DescribeGameSessions— Récupérez les métadonnées d'une session de jeu, y compris la durée d'un jeu et le nombre de joueurs actuels.

Configurer les files d'attente pour un placement optimal des sessions de jeu

Configurez des files d'attente sur plusieurs flottes et plusieurs régions pour placer des sessions de jeu avec les meilleures ressources d'hébergement disponibles en matière de coût, de latence et de résilience.

- CreateGameSessionQueue— Créez une file d'attente à utiliser lors du traitement des demandes de placement de sessions de jeu.
- DescribeGameSessionQueues— Récupère les files d'attente de sessions de jeu définies dans une GameLift région Amazon.
- UpdateGameSessionQueue— Modifie la configuration d'une file d'attente de sessions de jeu.
- DeleteGameSessionQueue— Supprime une file d'attente de sessions de jeu de la région.

Gérer les alias

Utilisez des alias pour représenter vos flottes ou créer une destination alternative de terminal. Les alias sont utiles lors de la transition de l'activité du jeu d'une flotte à une autre, par exemple lors des mises à jour des builds du serveur de jeu.

- CreateAlias— Définissez un nouvel alias et attribuez-le éventuellement à une flotte.
- ListAliases— Obtenez tous les alias de flotte définis dans une GameLift région Amazon.
- DescribeAlias— Récupère des informations sur un alias existant
- UpdateAlias— Modifiez les paramètres d'un alias, par exemple en le redirigeant d'une flotte vers une autre.
- DeleteAlias— Supprime un alias de la région
- ResolveAlias— Récupère l'ID de flotte vers lequel pointe un alias spécifié.

Accéder aux instances d'hébergement

Affichez des informations sur les instances individuelles d'une flotte ou demandez un accès à distance à une instance de flotte spécifiée pour le dépannage.

- DescribeInstances— Obtenez des informations sur chaque instance d'un parc, notamment l'ID de l'instance, l'adresse IP, l'emplacement et le statut.
- GetInstanceAccess— Demandez les identifiants d'accès nécessaires pour vous connecter à distance à une instance spécifiée d'un parc.

Configurer l'appairage de VPC

Créez et gérez des connexions d'appairage VPC entre vos ressources GameLift d'hébergement Amazon et d'autres AWS ressources.

- CreateVpcPeeringAuthorization— Autorisez une connexion de peering à l'un de vos VPC.
- DescribeVpcPeeringAuthorizations— Récupère les autorisations de connexion d'appairage valides.
- DeleteVpcPeeringAuthorization— Supprime une autorisation de connexion de peering.
- CreateVpcPeeringConnection— Établissez une connexion d'appairage entre le VPC d'une GameLift flotte Amazon et l'un de vos VPC.
- DescribeVpcPeeringConnections— Récupérez des informations sur les connexions d'appairage VPC actives ou en attente avec un parc AmazonGameLift.

- DeleteVpcPeeringConnection— Supprime une connexion d'appairage VPC avec une flotte AmazonGameLift.

Démarrez des sessions de jeu et rejoignez des joueurs

Appelez ces opérations depuis le service client de votre jeu pour démarrer de nouvelles sessions de jeu, obtenir des informations sur les sessions de jeu existantes et associer des joueurs à des sessions de jeu. Ces opérations sont destinées à être utilisées avec des serveurs de jeu personnalisés hébergés sur AmazonGameLift. Si vous utilisez des serveurs en temps réel, gérez les sessions de jeu à l'aide duRéférence de l'API client (C#) de Realtime Servers.

- Démarrage de nouvelles sessions de jeu pour un ou plusieurs joueurs.
  - StartGameSessionPlacement— Demandez GameLift à Amazon de trouver les meilleures ressources d'hébergement disponibles et de démarrer une nouvelle session de jeu. Il s'agit de la méthode préférée pour créer de nouvelles sessions de jeu. Il s'appuie sur les files d'attente des sessions de jeu pour suivre la disponibilité de l'hébergement dans plusieurs régions et utilise les algorithmes FleetIQ pour hiérarchiser les emplacements en fonction de la latence des joueurs, du coût d'hébergement, de la localisation, etc.
  - DescribeGameSessionPlacement— Obtenez les détails et le statut d'une demande de placement.
  - StopGameSessionPlacement— Annule une demande de placement.
  - CreateGameSession— Démarrez une nouvelle session de jeu vide sur un emplacement spécifique de la flotte. Cette opération vous permet de mieux contrôler où démarrer la session de jeu, au lieu d'utiliser FleetIQ pour évaluer les options de placement. Vous devez ajouter des joueurs à la nouvelle session de jeu lors d'une étape distincte.
- Faites participer les joueurs à des sessions de jeu existantes. Trouvez des sessions de jeu en cours grâce aux créneaux disponibles et réservez-les aux nouveaux joueurs.
  - CreatePlayerSession— Réservez un emplacement libre pour qu'un joueur puisse rejoindre une session de jeu.
  - CreatePlayerSessions— Réservez des emplacements libres pour que plusieurs joueurs puissent participer à une session de jeu.
- Utilisation avec des données de session de jeu et de session de joueur. Gérez les informations relatives aux sessions de jeu et aux sessions des joueurs.
  - SearchGameSessions— Demandez une liste des sessions de jeu actives en fonction d'un ensemble de critères de recherche.

- DescribeGameSessions— Récupérez des métadonnées pour des sessions de jeu spécifiques, y compris la durée d'activité et le nombre de joueurs actuels.
- DescribeGameSessionDetails— Récupérez les métadonnées, y compris le paramètre de protection des sessions de jeu, pour une ou plusieurs sessions de jeu.
- DescribePlayerSessions— Obtenez des détails sur l'activité des joueurs, notamment leur statut, leur temps de jeu et leurs données.
- UpdateGameSession— Modifiez les paramètres de session de jeu, tels que le nombre maximum de joueurs et la politique d'inscription.
- GetGameSessionLogUrl— Récupère l'emplacement des journaux enregistrés pour une session de jeu.

Référence des serveurs Amazon GameLift Realtime

Cette section contient de la documentation de référence pour le SDK Amazon GameLift Realtime Servers. Il inclut l'API client en temps réel ainsi que des instructions pour configurer votre script Realtime Servers.

Rubriques

- Référence de l'API client (C#) de Realtime Servers
- Référence de script Amazon GameLift Realtime Servers

Référence de l'API client (C#) de Realtime Servers

Utilisez l'API client en temps réel pour préparer vos clients de jeux multijoueurs à utiliser avec les serveurs Amazon GameLift Realtime. Pour plus d'informations sur le processus d'intégration, consultez Préparez votre serveur en temps réel. L'API client contient un ensemble d'appels d'API synchrones et de rappels asynchrones qui permettent à un client de jeu de se connecter à un serveur en temps réel et d'échanger des messages et des données avec d'autres clients de jeu via le serveur.

Cet API est défini dans les bibliothèques suivantes :

Client.cs

- Actions synchrones
- Rappels asynchrones
- Types de données

Pour configurer l'API du client en temps réel

1. Téléchargez le SDK du client Amazon GameLift Realtime.
2. Développez les bibliothèques SDK C#. Localisez le fichier de solution `GameLiftRealtimeClientSdkNet45.sln`. Consultez le fichier `README.md` relatif au kit SDK C# Server pour connaître la configuration minimale requise et les options de build supplémentaires. Chargez le fichier de solution dans un IDE. Pour générer les bibliothèques du SDK, restaurez les NuGet packages et créez la solution.
3. Ajoutez les bibliothèques du client en temps réel à votre projet de client de jeu.

Référence de l'API client (C#) de Realtime Servers : Actions

Cette référence d'API client C# Realtime peut vous aider à préparer votre jeu multijoueur en vue de son utilisation avec les serveurs en temps réel déployés sur les flottes Amazon. GameLift Pour plus d'informations sur le processus d'intégration, consultez Préparez votre serveur en temps réel.

- Actions synchrones
- Rappels asynchrones
- Types de données

Client()

Initialise un nouveau client pour communiquer avec le serveur Realtime et identifie le type de connexion à utiliser.

Syntaxe

```
public Client(ClientConfiguration configuration)
```

Paramètres

clientConfiguration

Détails de configuration spécifiant le type de connexion client/serveur. Vous pouvez choisir d'appeler Client() sans ce paramètre ; cependant, cette approche se traduit par une connexion non sécurisée par défaut.

Type: ClientConfiguration

Obligatoire : non

Valeur renvoyée

Renvoie une instance du client Realtime à utiliser avec la communication avec le serveur Realtime.

Connect()

Demande une connexion à un processus serveur qui héberge une session de jeu.

Syntaxe

```
public ConnectionStatus Connect(string endpoint, int remoteTcpPort, int listenPort,
 ConnectionToken token)
```

Paramètres

point de terminaison

Adresse IP ou nom DNS de la session de jeu à laquelle se connecter. Le point de terminaison est spécifié dans un `GameSession` objet, qui est renvoyé en réponse à un appel client aux actions de l'GameLiftAPI Amazon du AWS SDK StartGameSessionPlacementCreateGameSession, ou DescribeGameSessions.

> **Note**
>
> Si le serveur Realtime s'exécute sur une flotte avec un certificat TLS, vous devez utiliser le nom DNS.

Type : String

Obligatoire : oui

remoteTcpPort

Numéro de port pour la connexion TCP attribuée à la session de jeu. Ces informations sont spécifiées dans un `GameSession` objet, qui est renvoyé en réponse à une StartGameSessionPlacementCreateGameSessionDescribeGameSessiondemande ou.

Type : entier

Valeurs valides : 1900 à 2000.

Obligatoire : oui

listenPort

Numéro de port que le client de jeu surveille pour des messages envoyés à l'aide du canal UDP.

Type : entier

Valeurs valides : 33400 à 33500.

Obligatoire : oui

token

Informations facultatives qui identifient la demande client de jeu au processus serveur.

Type: ConnectionToken

Obligatoire : oui

Valeur renvoyée

Renvoie une valeur d'ConnectionStatusénumération indiquant l'état de connexion du client.

Disconnect()

Lorsque vous êtes connecté à une session de jeu, déconnecte le client de jeu à partir de la session de jeu.

Syntaxe

```
public void Disconnect()
```

Paramètres

Cette action n'a aucun paramètre.

Valeur renvoyée

Cette méthode ne renvoie rien.

NewMessage()

Crée un nouvel objet de message avec un code d'opération spécifié. Une fois qu'un message est renvoyé, complétez le contenu du message en spécifiant une cible, en mettant à jour le mode de remise, et en ajoutant une charge utile de données en fonction de vos besoins. Une fois terminé, envoyez le message à l'aide de `SendMessage()`.

Syntaxe

```
public RTMessage NewMessage(int opCode)
```

Paramètres

opCode

Code d'opération défini par le développeur qui identifie un événement ou une action de jeu, par exemple un mouvement de joueur ou une notification.

Type : entier

Obligatoire : oui

Valeur renvoyée

Renvoie un objet RTMessage contenant l'opération spécifiée par défaut et la méthode de livraison de code. Le paramètre d'intention d'acheminement est défini sur `FAST` par défaut.

SendMessage()

Envoie un message à un joueur ou un groupe à l'aide de la méthode de livraison spécifiée.

Syntaxe

```
public void SendMessage(RTMessage message)
```

Paramètres

message

Message d'objet qui spécifie le destinataire cible, la méthode de livraison et le contenu du message.

Type: RTMessage

Obligatoire : oui

Valeur renvoyée

Cette méthode ne renvoie rien.

JoinGroup()

Ajoute le joueur à un groupe spécifié. Les groupes peuvent contenir n'importe quel joueur connecté au jeu. Une fois qu'il a rejoint le serveur, le joueur reçoit tous les futurs messages envoyés au groupe et peut envoyer des messages à l'ensemble du groupe.

Syntaxe

```
public void JoinGroup(int targetGroup)
```

Paramètres

targetGroup

ID unique qui identifie le groupe auquel ajouter le joueur. Les ID de groupe sont définis par le développeur.

Type : entier

Obligatoire : oui

Valeur renvoyée

Cette méthode ne renvoie rien. Étant donné que cette demande est envoyée à l'aide de la méthode de livraison fiable (TCP), un échec de demande déclenche le rappel OnError().

LeaveGroup()

Retire le joueur d'un groupe spécifié. Une fois qu'il n'est plus dans le groupe, le joueur ne reçoit pas les messages envoyés au groupe et ne peut envoyer de messages à l'ensemble du groupe.

Syntaxe

```
public void LeaveGroup(int targetGroup)
```

Paramètres

targetGroup

ID unique qui identifie le groupe duquel retirer le joueur. Les ID de groupe sont définis par le développeur.

Type : entier

Obligatoire : oui

Valeur renvoyée

Cette méthode ne renvoie rien. Étant donné que cette demande est envoyée à l'aide de la méthode de livraison fiable (TCP), un échec de demande déclenche le rappel OnError().

RequestGroupMembership()

Demande qu'une liste de joueurs dans le groupe spécifié soit envoyée au client de jeu. Tout joueur peut demander ces informations, qu'il s'agisse ou non d'un membre du groupe. En réponse à cette demande, la liste d'adhésion est envoyée au client via un rappel OnGroupMembershipUpdated().

Syntaxe

```
public void RequestGroupMembership(int targetGroup)
```

Paramètres

targetGroup

ID unique qui identifie le groupe pour lequel obtenir des informations sur l'adhésion. Les ID de groupe sont définis par le développeur.

Type : entier

Obligatoire : oui

Valeur renvoyée

Cette méthode ne renvoie rien.

Référence de l'API client (C#) de Realtime Servers : rappels asynchrones

Utilisez cette référence d'API client C# Realtime pour vous aider à préparer votre jeu multijoueur en vue de son utilisation avec les serveurs en temps réel déployés sur les flottes Amazon. GameLift Pour plus d'informations sur le processus d'intégration, consultez Préparez votre serveur en temps réel.

- Actions synchrones
- Rappels asynchrones
- Types de données

Un client de jeu a besoin de mettre en œuvre ces méthodes de rappel pour répondre aux événements. Le serveur Realtime invoque ces rappels pour envoyer des informations relatives au jeu au client du jeu. Les rappels pour les mêmes événements peuvent également être implémentés avec une logique de jeu personnalisée dans le script du serveur en temps réel. Consultez Rappels de script pour les serveurs en temps réel.

Les méthodes de rappel sont définies dans `ClientEvents.cs`.

OnOpen()

Appelée lorsque le processus serveur accepte la demande de connexion d'un client de jeu et ouvre une connexion.

Syntaxe

```
public void OnOpen()
```

Paramètres

Cette méthode ne prend aucun paramètre.

Valeur renvoyée

Cette méthode ne renvoie rien.

OnClose()

Appelée lorsque le processus serveur met fin à la connexion avec le client de jeu, comme lorsqu'une session de jeu se termine.

Syntaxe

```
public void OnClose()
```

Paramètres

Cette méthode ne prend aucun paramètre.

Valeur renvoyée

Cette méthode ne renvoie rien.

OnError()

Appelée lorsqu'un échec se produit pour une demande d'API du client en temps réel. Ce rappel peut être personnalisé afin de prendre en charge une grande variété d'erreurs de connexion.

Syntaxe

```
private void OnError(byte[] args)
```

Paramètres

Cette méthode ne prend aucun paramètre.

Valeur renvoyée

Cette méthode ne renvoie rien.

OnDataReceived()

Invoqué lorsque le client du jeu reçoit un message du serveur Realtime. Il s'agit de la méthode principale de réception de messages et de notifications par un client de jeu.

Syntaxe

```
public void OnDataReceived(DataReceivedEventArgs dataReceivedEventArgs)
```

Paramètres

dataReceivedEventArgs

Informations liées à l'activité de messages.

Type: DataReceivedEventArgs

Obligatoire : oui

Valeur renvoyée

Cette méthode ne renvoie rien.

OnGroupMembershipUpdated()

Appelée lorsque l'adhésion à un groupe auquel le joueur appartient a été mise à jour. Ce rappel est également invoqué lorsqu'un client appelle `RequestGroupMembership`.

Syntaxe

```
public void OnGroupMembershipUpdated(GroupMembershipEventArgs groupMembershipEventArgs)
```

Paramètres

groupMembershipEventArgs

Informations liées à l'activité d'adhésion d'un groupe.

Type: GroupMembershipEventArgs

Obligatoire : oui

Valeur renvoyée

Cette méthode ne renvoie rien.

Référence de l'API client (C#) de Realtime Servers : types de données

Cette référence d'API client C# Realtime peut vous aider à préparer votre jeu multijoueur en vue de son utilisation avec les serveurs en temps réel déployés sur les flottes Amazon. GameLift Pour plus d'informations sur le processus d'intégration, consultez Préparez votre serveur en temps réel.

- Actions synchrones
- Rappels asynchrones

- Les types de données

ClientConfiguration

Informations sur la façon dont le client du jeu se connecte à un serveur en temps réel.

Table des matières

ConnectionType

Type de connexion client/serveur à utiliser, sécurisée ou non sécurisée. Si vous ne spécifiez pas de type de connexion, la valeur par défaut est non sécurisée.

> **Note**
>
> Lors de la connexion à un serveur en temps réel sur une flotte sécurisée avec un certificat TLS, vous devez utiliser la valeur RT_OVER_WSS_DTLS_TLS12.

Type : valeur `ConnectionType` enum .

Obligatoire : non

ConnectionToken

Informations concernant le client du jeu et/ou le joueur qui demande une connexion à un serveur en temps réel.

Table des matières

playerSessionId

Identifiant unique émis par Amazon GameLift lors de la création d'une nouvelle session de joueur. Un identifiant de session de joueur est spécifié dans un `PlayerSession` objet, qui est renvoyé en réponse à un appel du client aux actions de l'GameLiftAPI StartGameSessionPlacementCreateGameSession, DescribeGameSessionPlacement, ou DescribePlayerSessions.

Type : String

Obligatoire : oui

payload

Informations définies par le développeur à communiquer au serveur en temps réel lors de la connexion. Cela inclut toutes les données arbitraires qui peuvent être utilisées pour un mécanisme de connexion personnalisé. Par exemple, une charge utile peut fournir des informations d'authentification à traiter par le script du serveur en temps réel avant d'autoriser un client à se connecter.

Type : tableau d'octets

Obligatoire : non

RTMessage

Contenu et informations de livraison pour un message. Un message doit spécifier soit un joueur ou un groupe cible.

Table des matières

opCode

Code d'opération défini par le développeur qui identifie un événement ou une action de jeu, par exemple un mouvement de joueur ou une notification. Un code Op d'un message fournit le contexte pour les données utiles qui sont fournies. Le code d'opération des messages créés à l'aide de NewMessage () est déjà défini, mais il peut être modifié à tout moment.

Type : entier

Obligatoire : oui

targetPlayer

ID unique qui identifie le destinataire voulu du message envoyé. La cible peut être le serveur lui-même (en utilisant l'ID de serveur) ou un autre joueur (à l'aide d'un ID de joueur).

Type : entier

Obligatoire : non

targetGroup

ID unique qui identifie le groupe destinataire voulu du message envoyé. Les ID de groupe sont définis par le développeur.

Type : entier

Obligatoire : non

deliveryIntent

Indique s'il convient d'envoyer le message à l'aide de la connexion TCP fiable ou à l'aide du canal UDP rapide. Messages créés à l'aide de NewMessage().

Type : DeliveryIntent enum

Valeurs valides : RAPIDE | FIABLE

Obligatoire : oui

payload

Contenu du message. Ces informations sont structurées de telle manière à être traitées par le client de jeu en fonction du code d'opération connexe. Elles peuvent contenir des données d'état du jeu ou d'autres informations qui doivent être communiquées entre les clients de jeu ou entre un client de jeu et le serveur en temps réel.

Type : tableau d'octets

Obligatoire : non

DataReceivedEventArgs

Les données fournies avec un rappel OnDataReceived().

Table des matières

sender

ID unique qui identifie l'entité (ID de joueur ou ID de serveur) ayant transmis le message.

Type : entier

Obligatoire : oui

opCode

Code d'opération défini par le développeur qui identifie un événement ou une action de jeu, par exemple un mouvement de joueur ou une notification. Un code Op d'un message fournit le contexte pour les données utiles qui sont fournies.

Type : entier

Obligatoire : oui

data

Contenu du message. Ces informations sont structurées de telle manière à être traitées par le client de jeu en fonction du code d'opération connexe. Elles peuvent contenir des données d'état du jeu ou d'autres informations qui doivent être communiquées entre les clients de jeu ou entre un client de jeu et le serveur en temps réel.

Type : tableau d'octets

Obligatoire : non

GroupMembershipEventArgs

Les données fournies avec un rappel OnGroupMembershipUpdated().

Table des matières

sender

ID unique qui identifie le joueur qui a demandé une mise à jour d'adhésion au groupe.

Type : entier

Obligatoire : oui

opCode

Code d'opération défini par le développeur qui identifie un événement ou une action de jeu.

Type : entier

Obligatoire : oui

groupId

ID unique qui identifie le groupe destinataire voulu du message envoyé. Les ID de groupe sont définis par le développeur.

Type : entier

Obligatoire : oui

playerId

Liste des ID de joueurs qui sont actuellement membres du groupe spécifié.

Type : tableau integer

Obligatoire : oui

Enums

Les énumérations définies pour le SDK du client en temps réel sont définies comme suit :

ConnectionStatus

- CONNECTÉ — Le client du jeu est connecté au serveur en temps réel via une connexion TCP uniquement. Tous les messages sont envoyés via TCP ; indépendamment de l'intention de livraison.
- CONNECTED_SEND_FAST — Le client du jeu est connecté au serveur Realtime via une connexion TCP et UDP. Cependant, la capacité de recevoir des messages via UDP n'est pas encore vérifiée. Par conséquent, tous les messages envoyés au client de jeu utilisent TCP.
- CONNECTED_SEND_AND_RECEIVE_FAST — Le client du jeu est connecté au serveur Realtime via une connexion TCP et UDP. Le client de jeu peut envoyer et recevoir des messages à l'aide de TCP ou UDP.
- CONNECTING Le client du jeu a envoyé une demande de connexion et le serveur Realtime est en train de la traiter.
- DISCONNECTED_CLIENT_CALL — Le client du jeu a été déconnecté du serveur en temps réel en réponse à une demande du client du jeu. Disconnect()
- DÉCONNECTÉ — Le client du jeu a été déconnecté du serveur en temps réel pour une raison autre qu'un appel de déconnexion du client.

ConnectionType

- RT_OVER_WSS_DTLS_TLS12 — Type de connexion sécurisée.

  À utiliser avec les serveurs en temps réel qui s'exécutent sur un GameLift parc avec un certificat TLS généré. Lorsque vous utilisez une connexion sécurisée, le trafic TCP est chiffré à l'aide de TLS 1.2 et le trafic UDP est chiffré à l'aide de DTLS 1.2.
- RT_OVER_WS_UDP_UNSECURED — Type de connexion non sécurisé.
- RT_OVER_WEBSOCKET — Type de connexion non sécurisé. Cette valeur n'est plus préférée.

DeliveryIntent

- RAPIDE — Livré via un canal UDP.
- FIABLE — Fourni via une connexion TCP.

Référence de script Amazon GameLift Realtime Servers

Utilisez ces ressources pour créer une logique personnalisée dans vos scripts en temps réel.

Rubriques

- Rappels de script pour les serveurs en temps réel
- Interface de serveurs en temps réel

Rappels de script pour les serveurs en temps réel

Vous pouvez fournir une logique personnalisée pour répondre aux événements en implémentant ces rappels dans votre script en temps réel.

Init

Initialise le serveur en temps réel et reçoit une interface de serveur en temps réel.

Syntaxe

```
init(rtsession)
```

onMessage

Appelé lorsqu'un message reçu est envoyé au serveur.

Syntaxe

```
onMessage(gameMessage)
```

onHealthCheck

Appelé pour définir le statut d'intégrité de la session de jeu. Par défaut, le statut d'intégrité est sain (ou `true`). Ce rappel peut être implémenté pour effectuer des vérifications de l'état personnalisées et renvoyer un statut.

Syntaxe

```
onHealthCheck()
```

onStartGameSéance

Appelé lorsqu'une nouvelle session de jeu démarre, avec un objet de session de jeu transmis.

Syntaxe

```
onStartGameSession(session)
```

onProcessTerminate

Invoqué lorsque le processus du serveur est interrompu par le GameLift service Amazon. Il peut agir comme déclencheur pour quitter de façon nette la session de jeu. Il n'y a pas besoin d'appeler `processEnding()`.

Syntaxe

```
onProcessTerminate()
```

onPlayerConnect

Appelé lorsqu'un joueur demande une connexion et a réussi la validation initiale.

Syntaxe

```
onPlayerConnect(connectMessage)
```

onPlayerAccepted

Appelé lorsqu'une connexion de joueur est acceptée.

Syntaxe

```
onPlayerAccepted(player)
```

onPlayerDisconnect

Appelé lorsqu'un joueur se déconnecte de la session de jeu, que ce soit en envoyant une demande de déconnexion ou par d'autres moyens.

Syntaxe

```
onPlayerDisconnect(peerId)
```

onProcessStarted

Appelé lorsqu'un processus serveur est démarré. Ce rappel autorise le script à effectuer toutes les tâches personnalisées nécessaires pour préparer l'hébergement d'une session de jeu.

Syntaxe

```
onProcessStarted(args)
```

onSendToJoueur

Appelé lorsqu'un message est reçu sur le serveur d'un joueur pour être livré à un autre joueur. Ce processus s'exécute avant que le message soit livré.

Syntaxe

```
onSendToPlayer(gameMessage)
```

onSendToGroupe

Appelé lorsqu'un message est reçu sur le serveur d'un joueur pour être livré à un groupe. Ce processus s'exécute avant que le message soit livré.

Syntaxe

```
onSendToGroup(gameMessage))
```

onPlayerJoinGroupe

Appelé lorsqu'un joueur envoie une demande pour rejoindre un groupe.

Syntaxe

```
onPlayerJoinGroup(groupId, peerId)
```

onPlayerLeaveGroupe

Appelé lorsqu'un joueur envoie une demande pour quitter un groupe.

Syntaxe

```
onPlayerLeaveGroup(groupId, peerId)
```

Interface de serveurs en temps réel

Lorsqu'un script en temps réel s'initialise, une interface vers le serveur en temps réel est renvoyée. Cette rubrique décrit les propriétés et les méthodes disponibles via cette interface. Apprenez-en davantage sur la rédaction de scripts en temps réel et consultez un exemple de script détaillé dansCréation d'un script en temps réel.

L'interface en temps réel permet d'accéder aux objets suivants :

- séance
- joueur (player)
- message de jeu (gameMessage)
- configuration

Objet de session en temps réel

Utilisez ces méthodes pour accéder aux informations liées au serveur et effectuer des actions liées au serveur.

getPlayers()

Récupère la liste des ID de pair pour les joueurs actuellement connectés à la session de jeu. Renvoie un tableau d'objets joueur.

Syntaxe

```
rtSession.getPlayers()
```

broadcastGroupMembershipMettre à jour ()

Déclenche la livraison d'une mise à jour de la liste des membres d'un groupe de joueurs. Spécifiez l'appartenance à diffuser (groupIdToDiffusion) et le groupe auquel vous souhaitez recevoir la mise à jour (targetGroupId). Les identifiants de groupe doivent être un entier positif ou « -1 » pour indiquer tous les groupes. Consultez un Exemple de script de serveurs en temps réel exemple d'ID de groupe définis par l'utilisateur.

Syntaxe

```
rtSession.broadcastGroupMembershipUpdate(groupIdToBroadcast, targetGroupId)
```

getServerId()

Récupère l'identifiant de pair unique du serveur, qui est utilisé pour router les messages vers le serveur.

Syntaxe

```
rtSession.getServerId()
```

getAllPlayersGroupId()

Récupère l'ID de groupe pour le groupe par défaut qui contient tous les joueurs actuellement connectés à la session de jeu.

Syntaxe

```
rtSession.getAllPlayersGroupId()
```

processEnding()

Déclenche le serveur en temps réel pour mettre fin au serveur de jeu. Cette fonction doit être appelée depuis le script Realtime pour quitter correctement une session de jeu.

Syntaxe

```
rtSession.processEnding()
```

getGameSessionIdentifiant ()

Récupère l'ID unique de la session de jeu en cours d'exécution.

Syntaxe

```
rtSession.getGameSessionId()
```

getLogger()

Récupère l'interface pour la journalisation. Utilisez cette commande pour consigner les instructions qui seront capturées dans vos journaux de session de jeu. L'enregistreur prend en charge les instructions « info », « warn » et « error ». Par exemple : `logger.info("<string>")`.

Syntaxe

```
rtSession.getLogger()
```

sendMessage()

Envoie un message, créé à l'aide de `newTextGameMessage` ou depuis le serveur en temps réel`newBinaryGameMessage`, à un joueur destinataire via le canal UDP. Identifiez le destinataire à l'aide de l'ID de pair du joueur.

Syntaxe

```
rtSession.sendMessage(gameMessage, targetPlayer)
```

sendGroupMessage()

Envoie un message, créé à l'aide `newTextGameMessage` ou à partir du serveur en temps réel`newBinaryGameMessage`, à tous les joueurs d'un groupe de joueurs via le canal UDP. Les identifiants de groupe doivent être un entier positif ou « -1 » pour indiquer tous les groupes. Consultez un Exemple de script de serveurs en temps réel exemple d'ID de groupe définis par l'utilisateur.

Syntaxe

```
rtSession.sendGroupMessage(gameMessage, targetGroup)
```

sendReliableMessage()

Envoie un message, créé à l'aide de `newTextGameMessage` ou depuis le serveur en temps réel`newBinaryGameMessage`, à un joueur destinataire via le canal TCP. Identifiez le destinataire à l'aide de l'ID de pair du joueur.

Syntaxe

```
rtSession.sendReliableMessage(gameMessage, targetPlayer)
```

sendReliableGroupVotre message ()

Envoie un message, créé à l'aide `newTextGameMessage` ou à partir du serveur en temps réel`newBinaryGameMessage`, à tous les joueurs d'un groupe de joueurs via le canal TCP. Les identifiants de groupe doivent être un entier positif ou « -1 » pour indiquer tous les groupes. Consultez un Exemple de script de serveurs en temps réel exemple d'ID de groupe définis par l'utilisateur.

Syntaxe

```
rtSession.sendReliableGroupMessage(gameMessage, targetGroup)
```

newTextGameVotre message ()

Crée un nouveau message contenant du texte, à envoyer depuis le serveur aux destinataires des joueurs à l'aide SendMessage des fonctions. Le format du message est similaire au format utilisé dans le kit SDK de client en temps réel (voir RTMessage). Renvoie un objet `gameMessage`.

Syntaxe

```
rtSession.newTextGameMessage(opcode, sender, payload)
```

newBinaryGameVotre message ()

Crée un nouveau message contenant des données binaires, à envoyer depuis le serveur aux destinataires des joueurs à l'aide SendMessage des fonctions. Le format du message est similaire au format utilisé dans le kit SDK de client en temps réel (voir RTMessage). Renvoie un objet `gameMessage`.

Syntaxe

```
rtSession.newBinaryGameMessage(opcode, sender, binaryPayload)
```

Objet joueur

Accédez aux informations liées au joueur.

player.peerId

Identifiant unique attribué à un client de jeu lorsqu'il se connecte au serveur Realtime et rejoint la session de jeu.

joueur. playerSessionId

ID de session du joueur référencé par le client du jeu lorsqu'il s'est connecté au serveur Realtime et a rejoint la session de jeu.

Objet message de jeu

Utilisez ces méthodes pour accéder aux messages reçus par le serveur en temps réel. Les messages reçus des clients de jeu ont la structure RTMessage.

getPayloadAsTexte ()

Obtient la charge utile du message du jeu sous forme de texte.

Syntaxe

```
gameMessage.getPayloadAsText()
```

gameMessage.opcode

Code d'opération contenu dans un message.

gameMessage.payload

Charge utile contenue dans un message. Peut être de type texte ou binaire.

gameMessage.sender

ID de pair du client de jeu qui a envoyé un message.

gameMessage.reliable

Booléen indiquant si le message a été envoyé via TCP (true) ou UDP (false).

Objet de configuration

L'objet de configuration peut être utilisé pour remplacer les configurations par défaut.

Configuration. Nombre maximum de joueurs

Le nombre maximum de connexions client-serveur pouvant être acceptées parRealTimeServers.

La valeur par défaut est 32.

configuration. pingIntervalTime

Intervalle de temps en millisecondes pendant lequel le serveur tentera d'envoyer un ping à tous les clients connectés pour vérifier que les connexions sont saines.

La valeur par défaut est de 3 000 ms.

Référence GameLift du SDK pour serveurs Amazon

Cette section contient de la documentation de référence pour le SDK GameLift du serveur Amazon. Utilisez le SDK du serveur pour intégrer vos serveurs de jeu personnalisés afin de communiquer avec le GameLift service Amazon.

Rubriques

- Référence GameLift du SDK Amazon Server pour C++
- Référence GameLift du SDK Amazon Server pour C#
- Référence GameLift du SDK Amazon Server pour Go
- Référence GameLift du SDK Amazon Server pour Unreal Engine

Référence GameLift du SDK Amazon Server pour C++

Vous pouvez utiliser cette référence du SDK pour serveurs Amazon GameLift C++ pour vous aider à préparer votre jeu multijoueur en vue de son utilisation avec AmazonGameLift. Pour plus de détails sur le processus d'intégration, consultezAjoutez Amazon GameLift à votre serveur de jeu.

Rubriques

- Référence GameLift du SDK Amazon Server 5.x pour C++
- Référence du SDK 3.x pour serveur Amazon GameLift C++

Référence GameLift du SDK Amazon Server 5.x pour C++

Cette référence du SDK 5.x pour Amazon GameLift C++ Server peut vous aider à préparer votre jeu multijoueur en vue de son utilisation avec Amazon. GameLift Pour plus de détails sur le processus d'intégration, consultezAjoutez Amazon GameLift à votre serveur de jeu.

> **Note**
>
> Cette rubrique décrit l'API Amazon GameLift C++ que vous pouvez utiliser lorsque vous créez avec la bibliothèque standard C++ (`std`). Plus précisément, cette documentation s'applique au code que vous compilez à l'aide de `-DGAMELIFT_USE_STD=1` cette option.

Rubriques

- Référence GameLift du SDK Amazon Server (C++) 5.x : Actions
- Référence GameLift du SDK Amazon Server (C++) : types de données

Référence GameLift du SDK Amazon Server (C++) 5.x : Actions

Vous pouvez utiliser cette référence relative au SDK du serveur Amazon GameLift C++ pour vous aider à préparer votre jeu multijoueur en vue de son utilisation avec Amazon GameLift. Pour plus de détails sur le processus d'intégration, consultezAjoutez Amazon GameLift à votre serveur de jeu.

> **Note**
>
> Cette rubrique décrit l'API Amazon GameLift C++ que vous pouvez utiliser lorsque vous créez avec la bibliothèque standard C++ (`std`). Plus précisément, cette documentation s'applique au code que vous compilez avec l'`-DGAMELIFT_USE_STD=1`option.

Actions

- GetSdkVersion()
- InitSDK()
- InitSDK()
- ProcessReady()
- ProcessReadyAsync()
- ProcessEnding()
- ActivateGameSession()
- UpdatePlayerSessionCreationPolicy()
- GetGameSessionId()
- GetTerminationTime()

- AcceptPlayerSession()
- RemovePlayerSession()
- DescribePlayerSessions()
- StartMatchBackfill()
- StopMatchBackfill()
- GetComputeCertificate()
- GetFleetRoleCredentials()
- Détruire ()

GetSdkVersion()

Renvoie le numéro de version actuel du kit SDK intégré dans le processus serveur.

Syntaxe

```
Aws::GameLift::AwsStringOutcome Server::GetSdkVersion();
```

Valeur renvoyée

En cas de réussite, renvoie la version actuelle du kit SDK en tant qu'objet the section called "AwsStringOutcome". L'objet renvoyé inclut le numéro de version (exemple`5.0.0`). En cas d'échec, renvoie un message d'erreur.

Exemple

```
Aws::GameLift::AwsStringOutcome SdkVersionOutcome =
 Aws::GameLift::Server::GetSdkVersion();
```

InitSDK()

Initialise le GameLift SDK Amazon pour un parc EC2 géré. Appelez cette méthode au lancement, avant toute autre initialisation liée à Amazon GameLift . Cette méthode lit les paramètres du serveur depuis l'environnement hôte afin de configurer la communication entre le serveur et le GameLift service Amazon.

Syntaxe

```
Server::InitSDKOutcome Server::initSdkOutcome = InitSDK();
```

Valeur renvoyée

Renvoie un the section called “Dans les résultats de son SDK” objet qui indique si le processus serveur est prêt à être appeléProcessReady().

Exemple

```
//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()

Initialise le GameLift SDK Amazon pour une Anywhere flotte. Appelez cette méthode au lancement, avant toute autre initialisation liée à Amazon GameLift . Cette méthode nécessite des paramètres de serveur explicites pour configurer la communication entre le serveur et le GameLift service Amazon.

Syntaxe

```
Server::InitSDKOutcome Server::initSdkOutcome = InitSDK(serverParameters);
```

Paramètres

ServerParameters

Pour initialiser un serveur de jeu sur une GameLift Anywhere flotte Amazon, créez un `ServerParameters` objet avec les informations suivantes :

- URL de l'URL WebSocket utilisée pour se connecter à votre serveur de jeu.
- ID du processus utilisé pour héberger votre serveur de jeu.
- L'ID de l'ordinateur hébergeant les processus de votre serveur de jeu.
- L'ID de la GameLift flotte Amazon contenant votre GameLift Anywhere ordinateur Amazon.
- Le jeton d'autorisation généré par l' GameLift opération Amazon.

Valeur renvoyée

Renvoie un the section called “Dans les résultats de son SDK” objet qui indique si le processus serveur est prêt à être appeléProcessReady().

> **Note**
>
> Si les appels à échouent pour `InitSDK()` les builds de jeu déployés sur des flottes Anywhere, vérifiez le `ServerSdkVersion` paramètre utilisé lors de la création de la ressource de build. Vous devez définir explicitement cette valeur en fonction de la version du SDK du serveur utilisée. La valeur par défaut de ce paramètre est 4.x, ce qui n'est pas compatible. Pour résoudre ce problème, créez une nouvelle version et déployez-la sur une nouvelle flotte.

Exemple

GameLift AnywhereExemple 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()

Indique à Amazon GameLift que le processus du serveur est prêt à héberger des sessions de jeu. Appelez cette méthode après l'avoir invoquée. InitSDK() Cette méthode ne doit être appelée qu'une seule fois par processus.

Syntaxe

```
GenericOutcome ProcessReady(const Aws::GameLift::Server::ProcessParameters
&processParameters);
```

Paramètres

processParameters

Objet ProcessParameters communiquant les informations suivantes relatives au processus serveur :

- Noms des méthodes de rappel implémentées dans le code du serveur de jeu invoqué par le GameLift service Amazon pour communiquer avec le processus du serveur.
- Numéro de port sur lequel le processus serveur écoute.
- Chemin d'accès à tous les fichiers spécifiques à une session de jeu que vous souhaitez qu'Amazon capture et GameLift stocke.

Valeur renvoyée

Renvoie un résultat générique consistant en un succès ou un échec avec un message d'erreur.

Exemple

Cet exemple illustre les implémentations de l'appel ProcessReady() et de la fonction déléguée.

```
// 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()

Indique au GameLift service Amazon que le processus du serveur est prêt à héberger des sessions de jeu. Cette méthode doit être appelée une fois que le processus serveur est prêt à héberger une session de jeu. Les paramètres spécifient les noms des fonctions de rappel qu'Amazon GameLift doit appeler dans certaines circonstances. Le code du serveur de jeux doit implémenter ces fonctions.

Cet appel est asynchrone. Pour effectuer un appel synchrone, utilisez ProcessReady(). Pour plus d'informations, consultez Initialiser le processus du serveur.

Syntaxe

```
GenericOutcomeCallable ProcessReadyAsync(
    const Aws::GameLift::Server::ProcessParameters &processParameters);
```

Paramètres

processParameters

Objet ProcessParameters communiquant les informations suivantes relatives au processus serveur :

- Noms des méthodes de rappel implémentées dans le code du serveur de jeu invoqué par le GameLift service Amazon pour communiquer avec le processus du serveur.
- Numéro de port sur lequel le processus serveur écoute.
- Chemin d'accès à tous les fichiers spécifiques à une session de jeu que vous souhaitez qu'Amazon capture et GameLift stocke.

Obligatoire : oui

Valeur renvoyée

Renvoie un résultat générique consistant en un succès ou un échec avec un message d'erreur.

Exemple

```
// 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()

Indique à Amazon GameLift que le processus du serveur est en train de se terminer. Appelez cette méthode après toutes les autres tâches de nettoyage (y compris la fermeture de la session de jeu active) et avant de terminer le processus. En fonction du résultat de`ProcessEnding()`, le processus se termine avec succès (0) ou erreur (-1) et génère un événement de flotte. Si le processus se termine par une erreur, l'événement de flotte généré est`SERVER_PROCESS_TERMINATED_UNHEALTHY`.

Syntaxe

```
Aws::GameLift::GenericOutcome processEndingOutcome =
 Aws::GameLift::Server::ProcessEnding();
```

Valeur renvoyée

Renvoie un résultat générique consistant en un succès ou un échec avec un message d'erreur.

Exemple

Cet exemple appelle `ProcessEnding()` et `Destroy()` avant de terminer le processus du serveur avec un code de sortie de réussite ou d'erreur.

```
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()

Informe Amazon GameLift que le processus du serveur a activé une session de jeu et qu'il est désormais prêt à recevoir les connexions des joueurs. Cette action doit être appelée dans le cadre de la fonction de `onStartGameSession()` rappel, après toute initialisation de session de jeu.

Syntaxe

```
Aws::GameLift::GenericOutcome activateGameSessionOutcome =
 Aws::GameLift::Server::ActivateGameSession();
```

Valeur renvoyée

Renvoie un résultat générique consistant en un succès ou un échec avec un message d'erreur.

Exemple

Cet exemple montre l'`ActivateGameSession()`appel dans le cadre de la fonction de `onStartGameSession()` délégation.

```
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()

Met à jour la capacité de la session de jeu à accepter de nouvelles sessions de joueur. Une session de jeu peut être définie pour accepter ou refuser toutes les nouvelles sessions joueur.

Syntaxe

```
GenericOutcome
 UpdatePlayerSessionCreationPolicy(Aws::GameLift::Model::PlayerSessionCreationPolicy
 newPlayerSessionPolicy);
```

Paramètres

playerCreationSessionPolitique

Type : valeur `PlayerSessionCreationPolicy` enum.

Obligatoire : oui

Valeur renvoyée

Renvoie un résultat générique consistant en un succès ou un échec avec un message d'erreur.

Exemple

Cet exemple définit la stratégie de participation de la session de jeu actuelle de manière à ce que tous les joueurs soient acceptés.

```
Aws::GameLift::GenericOutcome outcome =

 Aws::GameLift::Server::UpdatePlayerSessionCreationPolicy(Aws::GameLift::Model::PlayerSessionCr
```

GetGameSessionId()

Récupère l'ID de la session de jeu hébergée par le processus serveur actif.

Pour les processus inactifs qui ne sont pas activés lors d'une session de jeu, l'appel renvoie unthe section called “GameLiftError”.

Syntaxe

```
AwsStringOutcome GetGameSessionId()
```

Paramètres

Cette action n'a aucun paramètre.

Valeur renvoyée

En cas de réussite, renvoie l'ID de session de jeu en tant qu'objet the section called “AwsStringOutcome”. En cas d'échec, renvoie un message d'erreur.

Pour les processus inactifs qui ne sont pas activés lors d'une session de jeu, l'appel renvoie `Success = True` et `GameSessionId =""`.

Exemple

```
Aws::GameLift::AwsStringOutcome sessionIdOutcome =
 Aws::GameLift::Server::GetGameSessionId();
```

GetTerminationTime()

Renvoie l'heure d'arrêt planifiée pour un processus serveur, si une heure de résiliation est disponible. Un processus serveur prend des mesures après avoir reçu un `onProcessTerminate()` rappel d'Amazon GameLift. Amazon GameLift appelle `onProcessTerminate()` pour les raisons suivantes :

- Lorsque le processus du serveur a signalé un mauvais état de santé ou n'a pas répondu à Amazon GameLift.
- Lorsque vous mettez fin à l'instance lors d'un événement de réduction de la taille.
- Lorsqu'une instance est interrompue en raison d'une interruption ponctuelle.

Syntaxe

```
AwsDateTimeOutcome GetTerminationTime()
```

Valeur renvoyée

En cas de succès, renvoie l'heure de fin sous forme d'`AwsDateTimeOutcome`objet. La valeur est le délai de résiliation, exprimé en ticks écoulés depuis. `0001 00:00:00` Par exemple, la valeur de la date et de l'heure `2020-09-13 12:26:40 -000Z` est égale à celle `637355968000000000` des ticks. Si aucune heure de résiliation n'est disponible, renvoie un message d'erreur.

Si le processus n'a pas reçu de ProcessParameters. OnProcessTerminate() rappel, un message d'erreur est renvoyé. Pour plus d'informations sur l'arrêt d'un processus serveur, consultezRépondre à une notification d'arrêt d'un processus serveur.

Exemple

```
Aws::GameLift::AwsLongOutcome TermTimeOutcome =
 Aws::GameLift::Server::GetTerminationTime();
```

AcceptPlayerSession()

Informe Amazon GameLift qu'un joueur possédant l'identifiant de session de joueur spécifié s'est connecté au processus du serveur et doit être validé. Amazon GameLift vérifie que l'identifiant de session du joueur est valide. Une fois la session du joueur validée, Amazon GameLift change le statut de l'emplacement du joueur de RÉSERVÉ à ACTIF.

Syntaxe

```
GenericOutcome AcceptPlayerSession(String playerSessionId)
```

Paramètres

playerSessionId

Identifiant unique émis par Amazon GameLift lors de la création d'une nouvelle session de joueur.

Valeur renvoyée

Renvoie un résultat générique consistant en un succès ou un échec avec un message d'erreur.

Exemple

Cet exemple traite une demande de connexion qui inclut la validation et le rejet d'identifiants de session de joueur non valides.

```
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()

Indique à Amazon GameLift qu'un joueur s'est déconnecté du processus du serveur. En réponse, Amazon GameLift modifie l'emplacement du joueur pour le rendre disponible.

Syntaxe

```
GenericOutcome RemovePlayerSession(String playerSessionId)
```

Paramètres

`playerSessionId`

Identifiant unique émis par Amazon GameLift lors de la création d'une nouvelle session de joueur.

Valeur renvoyée

Renvoie un résultat générique consistant en un succès ou un échec avec un message d'erreur.

Exemple

```
Aws::GameLift::GenericOutcome disconnectOutcome =
 Aws::GameLift::Server::RemovePlayerSession(playerSessionId);
```

DescribePlayerSessions()

Récupère les données de session du joueur, notamment les paramètres, les métadonnées de session et les données du joueur. Utilisez cette méthode pour obtenir des informations sur les points suivants :

- Une session solo
- Toutes les sessions des joueurs au cours d'une session de jeu
- Toutes les sessions de joueur associées à un identifiant de joueur unique

Syntaxe

```
DescribePlayerSessionsOutcome DescribePlayerSessions(DescribePlayerSessionsRequest
 describePlayerSessionsRequest)
```

Paramètres

DescribePlayerSessionsRequest

the section called “DescribePlayerSessionsRequest”Objet qui décrit les sessions de joueur à récupérer.

Valeur renvoyée

En cas de réussite, renvoie un objet the section called “DescribePlayerSessionsOutcome” qui contient un ensemble d'objets de session de joueur correspondant aux paramètres de la demande.

Exemple

Cet exemple demande toutes les sessions de joueur activement connectées à une session de jeu spécifiée. En omettant NextTokenet en définissant la valeur limite sur 10, Amazon GameLift renvoie les 10 premiers enregistrements de session de joueur correspondant à la demande.

```
// 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()

Envoie une demande de recherche de nouveaux joueurs pour des emplacements ouverts dans une session de jeu créée avec FlexMatch. Pour plus d'informations, voir la fonction de FlexMatch remblayage.

Cette action est asynchrone. Si de nouveaux joueurs sont jumelés, Amazon GameLift fournit des données de matchmaking mises à jour à l'aide de la fonction de rappel. `OnUpdateGameSession()`

Un processus de serveur ne peut comporter qu'une seule requête de renvoi de correspondance à la fois. Pour envoyer une nouvelle requête, appelez d'abord StopMatchBackfill() pour annuler la requête d'origine.

Syntaxe

```
StartMatchBackfillOutcome StartMatchBackfill (StartMatchBackfillRequest
 startBackfillRequest);
```

Paramètres

StartMatchBackfillRequest

Un StartMatchBackfillRequest objet qui communique les informations suivantes :

- ID de ticket à attribuer à la requête de renvoi. Ces informations sont facultatives ; si aucun identifiant n'est fourni, Amazon en GameLift générera un.
- Matchmaker auquel envoyer la requête. L'ARN de configuration complet est obligatoire. Cette valeur se trouve dans les données du matchmaker de la session de jeu.
- ID de la session de jeu à compléter.
- Les données de matchmaking disponibles pour les joueurs actuels de la session de jeu.

Valeur renvoyée

Renvoie un the section called “StartMatchBackfillOutcome” objet avec l'identifiant du ticket de remplacement correspondant, ou un échec avec un message d'erreur.

Exemple

```
// 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()

Annule une demande de remplacement de match active. Pour plus d'informations, voir la fonction de FlexMatch remblayage.

Syntaxe

```
GenericOutcome StopMatchBackfill (StopMatchBackfillRequest stopBackfillRequest);
```

Paramètres

StopMatchBackfillRequest

Un StopMatchBackfillRequest objet identifiant le ticket de matchmaking à annuler :

- L'identifiant du ticket attribué à la demande de remblayage.
- L'entremetteur à qui la demande de remblayage a été envoyée.
- La session de jeu associée à la demande de remplacement.

Valeur renvoyée

Renvoie un résultat générique consistant en un succès ou un échec avec un message d'erreur.

Exemple

```
// 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()

Récupère le chemin d'accès au certificat TLS utilisé pour chiffrer la connexion réseau entre votre ressource GameLift Anywhere informatique Amazon et Amazon. GameLift Vous pouvez utiliser le chemin du certificat lorsque vous enregistrez votre appareil informatique dans une GameLift Anywhere flotte Amazon. Pour plus d'informations, voir, RegisterCompute.

Syntaxe

```
GetComputeCertificateOutcome Server::GetComputeCertificate()
```

Valeur renvoyée

Retourne un the section called “GetComputeCertificateOutcome”.

Exemple

```
Aws::GameLift::GetComputeCertificateOutcome certificate =
 Aws::GameLift::Server::GetComputeCertificate();
```

GetFleetRoleCredentials()

Récupère les informations d'identification du rôle IAM qui autorisent Amazon GameLift à interagir avec d'autres. Services AWS Pour plus d’informations, consultez Communiquez avec les autres AWS ressources de vos flottes.

Syntaxe

```
GetFleetRoleCredentialsOutcome GetFleetRoleCredentials(GetFleetRoleCredentialsRequest
 request);
```

Paramètres

GetFleetRoleCredentialsRequest

Valeur renvoyée

Renvoie un objet the section called “GetFleetRoleCredentialsOutcome”.

Exemple

```
// 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);
```

Cet exemple montre l'utilisation de la `RoleSessionName` valeur facultative pour attribuer un nom à la session d'identification à des fins d'audit. Si vous ne fournissez pas de nom de session de rôle, la valeur par défaut « *`[fleet-id]`* - *`[host-id]`* » est utilisée.

```
// 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);
```

Détruire ()

Libère de la mémoire le SDK du serveur de GameLift jeu Amazon. Il est recommandé d'appeler cette méthode après `ProcessEnding()` et avant de terminer le processus. Si vous utilisez une flotte Anywhere et que vous n'interrompez pas les processus du serveur après chaque session de jeu, appelez `Destroy()` puis réinitialisez avant `InitSDK()` d'informer Amazon GameLift que le processus est prêt à héberger une session de jeu avec. `ProcessReady()`

Syntaxe

```
GenericOutcome Aws::GameLift::Server::Destroy();
```

Paramètres

Il n'y a aucun paramètre.

Valeur renvoyée

Renvoie un résultat générique consistant en un succès ou un échec avec un message d'erreur.

Exemple

```
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);
}
```

Référence GameLift du SDK Amazon Server (C++) : types de données

Vous pouvez utiliser cette référence relative au SDK du serveur Amazon GameLift C++ pour vous aider à préparer votre jeu multijoueur en vue de son utilisation avec Amazon GameLift. Pour plus de détails sur le processus d'intégration, consultezAjoutez Amazon GameLift à votre serveur de jeu.

> **Note**
>
> Cette rubrique décrit l'API Amazon GameLift C++ que vous pouvez utiliser lorsque vous créez avec la bibliothèque standard C++ (`std`). Plus précisément, cette documentation s'applique au code que vous compilez avec l'`-DGAMELIFT_USE_STD=1`option.

Types de données

- LogParameters
- ProcessParameters

- UpdateGameSession
- GameSession
- ServerParameters
- StartMatchBackfillRequest
- Joueur
- DescribePlayerSessionsRequest
- StopMatchBackfillRequest
- AttributeValue
- GetFleetRoleCredentialsRequest
- AwsLongOutcome
- AwsStringOutcome
- DescribePlayerSessionsOutcome
- DescribePlayerSessionsResult
- GenericOutcome
- GenericOutcomeCallable
- PlayerSession
- StartMatchBackfillOutcome
- StartMatchBackfillResult
- GetComputeCertificateOutcome
- GetComputeCertificateResult
- GetFleetRoleCredentialsOutcome
- GetFleetRoleCredentialsResult
- Dans les résultats de son SDK
- GameLiftError
- Enums

LogParameters

Un objet identifiant les fichiers générés au cours d'une session de jeu que vous souhaitez qu'Amazon télécharge et stocke GameLift à la fin de la session de jeu. Le serveur de jeu fournit `LogParameters` à Amazon dans GameLift le cadre d'un `ProcessParameters` objet lors d'un ProcessReady() appel.

Propriétés	Description
LogPaths	La liste des chemins de répertoire vers les fichiers journaux du serveur de jeu que vous souhaitez GameLift qu'Amazon stocke pour un accès futur. Le processus du serveur génère ces fichiers lors de chaque session de jeu. Vous définissez les chemins et les noms des fichiers sur votre serveur de jeu et vous les stockez dans le répertoire racine du jeu. Les chemins du journal doivent être absolus. Par exemple, si votre build de jeu stocke les journaux de session de jeu dans un chemin tel que`MyGame\sessionLogs\` , le chemin se `c:\game\MyGame\sessionLogs` trouve sur une instance Windows. Type : `std:vector<std::string>` Obligatoire : non

ProcessParameters

Ce type de données contient l'ensemble de paramètres envoyé à Amazon GameLift dans un fichierProcessReady().

Propriétés	Description
LogParameters	Objet avec des chemins de répertoire vers des fichiers générés au cours d'une session de jeu. Amazon GameLift copie et stocke les fichiers pour un accès futur. Type : `Aws::GameLift::Server:: LogParameters`

	Obligatoire : non
OnHealthCheck	Fonction de rappel GameLift invoquée par Amazon pour demander un rapport d'état de santé au processus du serveur. Amazon GameLift appelle cette fonction toutes les 60 secondes et attend une réponse pendant 60 secondes. Le processus du serveur revient `TRUE` s'il est sain, `FALSE` sinon sain. Si aucune réponse n'est renvoyée, Amazon considère GameLift que le processus du serveur n'est pas sain. Type : `std::function<bool()> onHealthCheck` Obligatoire : non
OnProcessTerminate	La fonction de rappel GameLift invoquée par Amazon pour forcer le processus du serveur à s'arrêter. Après avoir appelé cette fonction, Amazon GameLift attend 5 minutes que le processus du serveur s'arrête et répond par un ProcessEnding() appel avant d'arrêter le processus du serveur. Type : `std::function<void()> onProcessTerminate` Obligatoire : oui
OnRefreshConnection	Nom de la fonction de rappel GameLift invoquée par Amazon pour actualiser la connexion avec le serveur de jeu. Type : `void OnRefreshConnectionDelegate()` Obligatoire : oui

OnStartGameSession	La fonction de rappel GameLift invoquée par Amazon pour activer une nouvelle session de jeu. Amazon GameLift appelle cette fonction en réponse à une demande d'un client CreateGameSession. La fonction de rappel transmet un GameSessionobjet, tel que défini dans le Amazon GameLift API Reference. Type : `const std::function<void (Aws::GameLift::Model::GameSession)> onStartGameSession` Obligatoire : oui
OnUpdateGameSession	Fonction de rappel GameLift invoquée par Amazon pour transmettre un objet de session de jeu mis à jour au processus du serveur. Amazon GameLift appelle cette fonction lorsqu'une demande de mise en correspondance a été traitée pour fournir des données de matchmaking mises à jour. Il transmet un GameSessionobjet, une mise à jour de statut (`updateReason`) et l'identifiant du ticket de remplacement des matchs. Type : `std::function<void(Aws::GameLift::Server::Model::UpdateGameSession)> onUpdateGameSession` Obligatoire : non

Port	Numéro de port sur lequel le processus serveur écoute les connexions des nouveaux joueurs. La valeur doit être comprise dans la plage de ports configurée pour toutes les flottes déployant cette version de génération du serveur de jeux. Ce numéro de port est inclus dans les objets de session de jeu et de session de joueur, que les sessions de jeu utilisent pour se connecter à un processus serveur. Type : `Integer` Obligatoire : oui

UpdateGameSession

Ce type de données est mis à jour vers un objet de session de jeu, qui inclut la raison pour laquelle la session de jeu a été mise à jour et l'identifiant du ticket de remplissage associé si le remplissage est utilisé pour remplir les sessions des joueurs pendant la session de jeu.

Propriétés	Description
GameSession	Un GameSessionobjet défini par l' GameLift API Amazon. L'`GameSession` objet contient des propriétés décrivant une session de jeu. Type : `Aws::GameLift::Server::GameSession` Obligatoire : oui
UpdateReason	La raison pour laquelle la session de jeu est mise à jour. Type : `Aws::GameLift::Server::UpdateReason` Obligatoire : oui

Propriétés	Description
BackfillTicketId	L'identifiant du ticket de remplacement qui tente de mettre à jour la session de jeu. Type : `std::string` Obligatoire : non

GameSession

Ce type de données fournit les détails d'une session de jeu.

Propriétés	Description
GameSessionId	Identifiant unique pour la session de jeu. L'ARN d'une session de jeu a le format suivant :`arn:aws:gamelift:<region>::gamesession/<fleet ID>/<custom ID string or idempotency token>` . Type : `std::string` Obligatoire : non
Nom	Libellé descriptif de la session de jeu. Type : `std::string` Obligatoire : non
FleetId	Identifiant unique de la flotte sur laquelle s'exécute la session de jeu. Type : `std::string` Obligatoire : non
MaximumPlayerSessionCount	Le nombre maximum de connexions de joueurs à la session de jeu.

Propriétés	Description
	Type : `int` Obligatoire : non
Port	Le numéro de port de la session de jeu. Pour se connecter à un serveur de GameLift jeu Amazon, une application a besoin à la fois de l'adresse IP et du numéro de port. Type : `in` Obligatoire : non
IpAddress	Adresse IP de la session de jeu. Pour se connecter à un serveur de GameLift jeu Amazon, une application a besoin à la fois de l'adresse IP et du numéro de port. Type : `std::string` Obligatoire : non
GameSessionData	Ensemble de propriétés de session de jeu personnalisées, mises en forme en tant que valeur de chaîne unique. Type : `std::string` Obligatoire : non

Propriétés	Description
MatchmakerData	Informations sur le processus de matchmaking utilisé pour créer la session de jeu, en syntaxe JSON, formatée sous forme de chaîne. En plus de la configuration de matchmaking utilisée, il contient des données sur tous les joueurs affectés au match, y compris les attributs des joueurs et les affectations des équipes. Type : `std::string` Obligatoire : non
GameProperties	Ensemble de propriétés personnalisées pour une session de jeu, formatées sous forme de paires clé:valeur. Ces propriétés sont transmises avec une demande de démarrage d'une nouvelle session de jeu. Type : `std :: vector < GameProperty >` Obligatoire : non

Propriétés	Description
DnsName	Identifiant DNS attribué à l'instance qui exécute la session de jeu. Les valeurs ont le format suivant : • Flottes compatibles TLS :. `<unique identifier>.<region identifier>.amazongamelift.com` • Flottes non compatibles TLS :. `ec2-<unique identifier>.compute.amazonaws.com` Lorsque vous vous connectez à une session de jeu exécutée sur une flotte compatible TLS, vous devez utiliser le nom DNS et non l'adresse IP. Type : `std::string` Obligatoire : non

ServerParameters

Informations utilisées pour maintenir la connexion entre le serveur de jeu d'une GameLift Anywhere flotte Amazon et le GameLift service Amazon. Ces informations sont utilisées lors du lancement de nouveaux processus serveur avecInitSDK(). Pour les serveurs hébergés sur des instances EC2 GameLift gérées par Amazon, utilisez un objet vide.

Propriétés	Description
webSocketUrl	L'`GameLiftServerSdkEndpoint` Amazon GameLift revient lorsque vous `RegisterCompute` recherchez une ressource GameLift Anywhere informatique Amazon. Type : `std::string`

Propriétés	Description
	Obligatoire : oui
Identifiant du processus	Un identifiant unique enregistré auprès du processus serveur hébergeant votre jeu. Type : `std::string` Obligatoire : oui
hostId	Le `HostID` est celui `ComputeName` utilisé lorsque vous avez enregistré votre ordinateur. Pour plus d'informations, voir, RegisterCompute. Type : `std::string` Obligatoire : oui
ID de flotte	Identifiant unique de la flotte dans laquelle le calcul est enregistré. Pour plus d'informations, voir, RegisterCompute. Type : `std::string` Obligatoire : oui
Jeton d'authentification	Le jeton d'authentification généré par Amazon GameLift qui authentifie votre serveur auprès d'Amazon GameLift. Pour plus d'informations, voir, GetComputeAuthToken. Type : `std::string` Obligatoire : oui

StartMatchBackfillRequest

Informations utilisées pour créer une demande de remplissage par matchmaking. Le serveur de jeu communique ces informations à Amazon GameLift lors d'un StartMatchBackfill() appel.

Propriétés	Description
GameSessionArn	Un identifiant de session de jeu unique. L'opération API `GetGameSessionId` renvoie l'identifiant au format ARN. Type : `std::string` Obligatoire : oui
MatchmakingConfigurationArn	Un identifiant unique, sous la forme d'un ARN, que le système de jumelage doit utiliser pour cette demande. L'ARN du système de matchmaking correspondant à la session de jeu d'origine se trouve dans l'objet de session de jeu dans la propriété des données du système de matchmaking. Pour en savoir plus sur les données du système de jumelage, consultez la section Travailler avec les données du système de jumelage. Type : `std::string` Obligatoire : oui
Joueurs	Ensemble de données représentant tous les joueurs participant à la session de jeu. Le matchmaker utilise ces informations pour rechercher de nouveaux joueurs qui constitue nt de bonnes correspondances pour les joueurs actuels. Type : `std::vector<Player>`

Propriétés	Description
	Obligatoire : oui
TicketId	Un identifiant unique pour un ticket de demande de matchmaking ou de remplacement de match. Si vous ne fournissez aucune valeur, Amazon en GameLift génère une. Utilisez cet identifiant pour suivre l'état du ticket de renvoi de correspondance ou annuler la requête si nécessaire. Type : `std::string` Obligatoire : non

Joueur

Ce type de données représente un joueur dans le matchmaking. Lors du lancement d'une demande de matchmaking, un joueur dispose d'un identifiant de joueur, d'attributs et éventuellement de données de latence. Amazon GameLift ajoute les informations de l'équipe une fois le match terminé.

Propriétés	Description
LatencyInMS	Ensemble de valeurs exprimées en milliseco ndes qui indiquent le niveau de latence ressenti par un joueur lorsqu'il est connecté à un lieu. Si cette propriété est utilisée, le joueur n'est jumelé qu'aux emplacements répertoriés. Si un matchmaker dispose d'une règle qui évalue la latence, les joueurs doivent indiquer la latence pour être mis en relation. Type : `Dictionary<string,int>` Obligatoire : non

Propriétés	Description
PlayerAttributes	Une collection de paires clé:valeur contenant des informations sur les joueurs à utiliser dans le matchmaking. Les clés d'attribut du joueur doivent correspondre à PlayerAttributes celles utilisées dans un ensemble de règles de matchmaking. Pour plus d'informations sur les attributs des joueurs, consultez AttributeValue. Type : `std::map<std::string,AttributeValue>` Obligatoire : non
PlayerId	Identifiant unique pour un joueur. Type : `std::string` Obligatoire : non
Equipe	Le nom de l'équipe à laquelle le joueur est affecté lors d'un match. Vous définissez le nom de l'équipe dans le jeu de règles de matchmaking. Type : `std::string` Obligatoire : non

DescribePlayerSessionsRequest

Objet qui indique les sessions de joueur à récupérer. Le processus du serveur fournit ces informations lors d'un DescribePlayerSessions() appel à Amazon GameLift.

Propriétés	Description
GameSessionId	Un identifiant de session de jeu unique. Utilisez ce paramètre pour demander toutes les sessions de joueur pour la session de jeu spécifiée. Le format de l'identifiant de session de jeu est`arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string>`. `GameSessionID` Il s'agit d'une chaîne d'identification personnalisée ou d'un Type : `std::string` Obligatoire : non
PlayerSessionId	Identifiant unique d'une session de joueur. Utilisez ce paramètre pour demander une session de joueur spécifique. Type : `std::string` Obligatoire : non
PlayerId	L'identifiant unique d'un joueur. Utilisez ce paramètre pour demander toutes les sessions de joueur pour un joueur spécifique. veuillez consulter Générer des identifiants de joueurs. Type : `std::string` Obligatoire : non
PlayerSessionStatusFilter	État de la session du joueur sur lequel filtrer les résultats. Les statuts possibles des sessions de joueur sont les suivants : • RÉSERVÉ — La demande de session du joueur a été reçue, mais le joueur ne s'est

Propriétés	Description
	pas connecté au processus du serveur ni n'a été validé. • ACTIF — Le joueur a été validé par le processus du serveur et est connecté. • TERMINÉ — La connexion du joueur a été interrompue. • TIMEDOUT — Une demande de session de joueur a été reçue, mais le joueur ne s'est pas connecté ou n'a pas été validée dans le délai imparti (60 secondes). Type : `std::string` Obligatoire : non
NextToken	Le jeton indiquant le début de la page de résultats suivante. Pour spécifier le début du jeu de résultats, ne fournissez aucune valeur. Si vous fournissez un identifiant de session de joueur, ce paramètre est ignoré. Type : `std::string` Obligatoire : non
Limite	Nombre maximal de résultats à renvoyer. Si vous fournissez un identifiant de session de joueur, ce paramètre est ignoré. Type : `int` Obligatoire : non

StopMatchBackfillRequest

Informations utilisées pour annuler une demande de remplissage par matchmaking. Le serveur de jeu communique ces informations au GameLift service Amazon lors d'un StopMatchBackfill() appel.

Propriétés	Description
GameSessionArn	Identifiant de session de jeu unique de la demande annulée. Type : `char[]` Obligatoire : non
MatchmakingConfigurationArn	Un identifiant unique du matchmaker auquel cette demande a été envoyée. Type : `char[]` Obligatoire : non
TicketId	Identifiant unique du ticket de demande de remblayage à annuler. Type : `char[]` Obligatoire : non

AttributeValue

Utilisez ces valeurs dans les paires Joueur clé-valeur d'attribut. Cet objet vous permet de spécifier une valeur d'attribut à l'aide de n'importe quel type de données valide : chaîne, nombre, tableau de chaînes ou mappage de données. Chaque `AttributeValue` objet doit utiliser exactement l'une des propriétés disponibles : `SN`,`SL`, ou`SDM`.

Propriétés	Description
AttrType	Spécifie le type de valeur d'attribut. Les types de valeurs d'attributs possibles sont les suivants :

Propriétés	Description
	• NONE • CORDE • DOUBLE • LISTE_CHAÎNES • STRING_DOUBLE_MAP Obligatoire : non
S	Représente une valeur d'attribut de chaîne. Type : `std::string` Obligatoire : non
N	Représente une valeur d'attribut numérique. Type : `double` Obligatoire : non
SL	Représente un tableau de valeurs d'attributs de chaîne. Type : `std::vector<std::string>` Obligatoire : non
SDM	Représente un dictionnaire de clés de chaîne et de valeurs doubles. Type : `std::map<std::string, double>` Obligatoire : non

GetFleetRoleCredentialsRequest

Ce type de données donne au serveur de jeu un accès limité à vos autres AWS ressources. Pour plus d'informations, consultez Configurer un rôle de service IAM pour Amazon GameLift.

Propriétés	Description
RoleArn	Le nom de ressource Amazon (ARN) du rôle de service qui étend un accès limité à vos AWS ressources. Type : `std::string` Obligatoire : non
RoleSessionName	Nom de session de rôle que vous pouvez utiliser pour identifier une AWS Security Token Service AssumeRolesession de manière unique. Ce nom est exposé dans les journaux d'audit tels que ceux contenus dans CloudTrail. Type : `std::string` Obligatoire : non

AwsLongOutcome

Ce type de données résulte d'une action et produit un objet doté des propriétés suivantes :

Propriétés	Description
Résultat	Le résultat de l'action. Type : `long` Obligatoire : non
ResultWithOwnership	Le résultat de l'action, converti en valeur rvalue, afin que le code appelant puisse s'approprier l'objet.

Propriétés	Description
	Type : `long&&` Obligatoire : non
Réussite	Que l'action ait été couronnée de succès ou non. Type : `bool` Obligatoire : oui
Erreur	L'erreur qui s'est produite en cas d'échec de l'action. Type : the section called “GameLiftError” Obligatoire : non

AwsStringOutcome

Ce type de données résulte d'une action et produit un objet doté des propriétés suivantes :

Propriétés	Description
Résultat	Le résultat de l'action. Type : `std::string` Obligatoire : non
ResultWithOwnership	Le résultat de l'action, converti en valeur rvalue, afin que le code appelant puisse s'approprier l'objet. Type : `long&&` Obligatoire : non

Propriétés	Description
Réussite	Que l'action ait été couronnée de succès ou non. Type : `bool` Obligatoire : oui
Erreur	L'erreur qui s'est produite en cas d'échec de l'action. Type : the section called “GameLiftError” Obligatoire : non

DescribePlayerSessionsOutcome

Ce type de données résulte d'une action et produit un objet doté des propriétés suivantes :

Propriétés	Description
Résultat	Le résultat de l'action. Type : the section called “DescribePlayerSessionsResult” Obligatoire : non
ResultWithOwnership	Le résultat de l'action, converti en valeur rvalue, afin que le code appelant puisse s'approprier l'objet. Type : `Aws::GameLift::Server::Model::DescribePlayerSessionsResult&&` Obligatoire : non

Propriétés	Description
Réussite	Que l'action ait été couronnée de succès ou non. Type : `bool` Obligatoire : oui
Erreur	L'erreur qui s'est produite en cas d'échec de l'action. Type : the section called “GameLiftError” Obligatoire : non

DescribePlayerSessionsResult

Collection d'objets contenant les propriétés de chaque session de joueur correspondant à la demande.

Propriétés	Description
NextToken	Un jeton qui indique le début de la prochaine page séquentielle de résultats. Utilisez le jeton renvoyé lors d'un appel précédent à cette opération. Pour commencer au début du jeu de résultats, ne spécifiez aucune valeur. Si un ID de session de joueur est spécifié, ce paramètre est ignoré. Type : `std::string` Obligatoire : oui
PlayerSessions	Type : `IList<`the section called “PlayerSession” `>` Obligatoire :

Propriétés	Description
ResultWithOwnership	Le résultat de l'action, converti en valeur rvalue, afin que le code appelant puisse s'approprier l'objet. Type : `std::string&&` Obligatoire : non
Réussite	Que l'action ait été couronnée de succès ou non. Type : `bool` Obligatoire : oui
Erreur	L'erreur qui s'est produite en cas d'échec de l'action. Type : the section called “GameLiftError” Obligatoire : non

GenericOutcome

Ce type de données résulte d'une action et produit un objet doté des propriétés suivantes :

Propriétés	Description
Réussite	Que l'action ait été couronnée de succès ou non. Type : `bool` Obligatoire : oui
Erreur	L'erreur qui s'est produite en cas d'échec de l'action.

Propriétés	Description
	Type : the section called “GameLiftError” Obligatoire : non

GenericOutcomeCallable

Ce type de données est un résultat générique asynchrone. Il comprend les propriétés suivantes :

Propriétés	Description
Réussite	Que l'action ait été couronnée de succès ou non. Type : `bool` Obligatoire : oui
Erreur	L'erreur qui s'est produite en cas d'échec de l'action. Type : the section called “GameLiftError” Obligatoire : non

PlayerSession

Ce type de données représente une session de joueur qu'Amazon GameLift transmet au serveur de jeu. Pour plus d'informations, consultez PlayerSession.

Propriétés	Description
CreationTime	Type : `long` Obligatoire : non
FleetId	Type : `std::string`

Propriétés	Description
	Obligatoire : non
GameSessionId	Type : `std::string` Obligatoire : non
IpAddress	Type : `std::string` Obligatoire : non
PlayerData	Type : `std::string` Obligatoire : non
PlayerId	Type : `std::string` Obligatoire : non
PlayerSessionId	Type : `std::string` Obligatoire : non
Port	Type : `int` Obligatoire : non

Propriétés	Description
Statut	État de session de joueur pour filtrer les résultats. Lorsqu'un PlayerSessionId ou PlayerId est fourni, cela n' PlayerSessionStatusFilter a aucun effet sur la réponse. Type : Une `PlayerSessionStatus` énumération. Les valeurs possibles sont notamment les suivantes : • ACTIF • TERMINÉ • NOT_SET • RÉSERVÉ • DÉLAI EXPIRÉ Obligatoire : non
TerminationTime	Type : `long` Obligatoire : non
DnsName	Type : `std::string` Obligatoire : non

StartMatchBackfillOutcome

Ce type de données résulte d'une action et produit un objet doté des propriétés suivantes :

Propriétés	Description
Résultat	Le résultat de l'action. Type : the section called “StartMatchBackfillResult”

Propriétés	Description
	Obligatoire : non
ResultWithOwnership	Le résultat de l'action, converti en valeur rvalue, afin que le code appelant puisse s'approprier l'objet. Type : `StartMatchBackfillResult&&` Obligatoire : non
Réussite	Que l'action ait été couronnée de succès ou non. Type : `bool` Obligatoire : oui
Erreur	L'erreur qui s'est produite en cas d'échec de l'action. Type : the section called “GameLiftError” Obligatoire : non

StartMatchBackfillResult

Ce type de données résulte d'une action et produit un objet doté des propriétés suivantes :

Propriétés	Description
TicketId	Un identifiant unique pour un ticket de matchmaking. Si aucun identifiant de ticket n'est spécifié ici, Amazon en GameLift générera un sous la forme d'un UUID. Utilisez cet identifiant pour suivre l'état du ticket de remplacement des matchs et récupérer les résultats des matchs.

Propriétés	Description
	Type : `std::string` Obligatoire : non

GetComputeCertificateOutcome

Ce type de données résulte d'une action et produit un objet doté des propriétés suivantes :

Propriétés	Description
Résultat	Le résultat de l'action. Type : the section called “GetComputeCertificateResult” Obligatoire : non
ResultWithOwnership	Le résultat de l'action, converti en valeur rvalue, afin que le code appelant puisse s'approprier l'objet. Type : `Aws::GameLift::Server::Model::GetComputeCertificateResult&&` Obligatoire : non
Réussite	Que l'action ait été couronnée de succès ou non. Type : `bool` Obligatoire : oui
Erreur	L'erreur qui s'est produite en cas d'échec de l'action. Type : the section called “GameLiftError”

Propriétés	Description
	Obligatoire : non

GetComputeCertificateResult

Le chemin d'accès au certificat TLS sur votre ordinateur et le nom d'hôte du calcul.

Propriétés	Description
CertificatePath	Le chemin d'accès au certificat TLS sur votre ressource de calcul. Lorsque vous utilisez une flotte GameLift gérée par Amazon, ce chemin contient : • `certificate.pem` : le certificat de l'utilisateur final. La chaîne de certificats complète est la combinaison des `certificateChain.pem` éléments ajoutés à ce certificat. • `certificateChain.pem` : chaîne de certificats qui contient le certificat racine et les certificats intermédiaires. • `rootCertificate.pem` : le certificat racine. • `privateKey.pem` : clé privée pour le certificat d'utilisateur final. Type : `std::string` Obligatoire : non
ComputeName	Le nom de votre ressource de calcul. Type : `std::string` Obligatoire : non

GetFleetRoleCredentialsOutcome

Ce type de données résulte d'une action et produit un objet doté des propriétés suivantes :

Propriétés	Description
Résultat	Le résultat de l'action. Type : the section called “GetFleetRoleCredentialsResult” Obligatoire : non
ResultWithOwnership	Le résultat de l'action, converti en valeur rvalue, afin que le code appelant puisse s'approprier l'objet. Type : `Aws::GameLift::Server::Model::GetFleetRoleCredentialsResult` Obligatoire : non
Réussite	Que l'action ait été couronnée de succès ou non. Type : `bool` Obligatoire : oui
Erreur	L'erreur qui s'est produite en cas d'échec de l'action. Type : the section called “GameLiftError” Obligatoire : non

GetFleetRoleCredentialsResult

Propriétés	Description
AccessKeyId	ID de clé d'accès permettant d'authentifier et de fournir un accès à vos AWS ressources. Type : `string` Obligatoire : non
AssumedRoleId	ID de l'utilisateur auquel appartient le rôle de service. Type : `string` Obligatoire : non
AssumedRoleUserArn	Le nom de ressource Amazon (ARN) de l'utilisateur auquel appartient le rôle de service. Type : `string` Obligatoire : non
Expiration	Durée avant l'expiration de vos informations d'identification de session. Type : `DateTime` Obligatoire : non
SecretAccessKey	ID de clé d'accès secrète pour l'authentification. Type : `string` Obligatoire : non
SessionToken	Un jeton pour identifier la session active en cours qui interagit avec vos AWS ressources. Type : `string`

Propriétés	Description
	Obligatoire : non
Réussite	Que l'action ait été couronnée de succès ou non. Type : `bool` Obligatoire : oui
Erreur	L'erreur qui s'est produite en cas d'échec de l'action. Type : the section called “GameLiftError” Obligatoire : non

Dans les résultats de son SDK

> **Note**
>
> `InitSDKOutcome`n'est renvoyé que lorsque vous créez le SDK avec l'`std`indicateur. Si vous construisez avec le `nostd` drapeau, il the section called “GenericOutcome” est renvoyé à la place.

Propriétés	Description
Réussite	Que l'action ait été couronnée de succès ou non. Type : `bool` Obligatoire : oui
Erreur	L'erreur qui s'est produite en cas d'échec de l'action. Type : the section called “GameLiftError”

Propriétés	Description
	Obligatoire : non

GameLiftError

Propriétés	Description
ErrorType	Type d'erreur. Type : Une `GameLiftErrorType` énumération. Obligatoire : non
ErrorName	Nom de l'erreur. Type : `std::string` Obligatoire : non
ErrorMessage	Message d’erreur. Type : `std::string` Obligatoire : non

Enums

Les énumérations définies pour le SDK Amazon GameLift Server (C++) sont définies comme suit :

GameLiftErrorType

Valeur de chaîne indiquant le type d'erreur. Les valeurs valides sont les suivantes :

- MAUVAISE DEMANDE D'EXCEPTION
- GAMESESSION_ID_NOT_SET — L'ID de session de jeu n'a pas été défini.
- EXCEPTION DE SERVICE INTERNE
- LOCAL_CONNECTION_FAILED — La connexion locale à Amazon a échoué. GameLift

- NETWORK_NOT_INITIALIZED — Le réseau n'a pas été initialisé.
- SERVICE_CALL_FAILED — Un appel à un AWS service a échoué.
- DÉFAILLANCE DE CONNEXION AU SOCKET WEB
- WEBSOCKET_CONNECT_FAILURE_FORBIDDEN
- WEBSOCKET_CONNECT_FAILURE_INVALID_URL
- WEBSOCKET_CONNECT_FAILURE_TIMEOUT
- ALREADY_INITIALIZED — Le GameLift serveur ou le client Amazon a déjà été initialisé avec Initialize ().
- FLEET_MISMATCH — La flotte cible ne correspond pas à la flotte d'une GameSession ou d'une PlayerSession.
- GAMELIFT_CLIENT_NOT_INITIALIZED — Le client Amazon n'a pas été initialisé. GameLift
- GAMELIFT_SERVER_NOT_INITIALIZED — Le serveur Amazon n'a pas été initialisé. GameLift
- GAME_SESSION_ENDED_FAILED — Le SDK GameLift Amazon Server n'a pas pu contacter le service pour signaler la fin de la session de jeu.
- GAME_SESSION_NOT_READY — La session de jeu GameLift Amazon Server n'a pas été activée.
- GAME_SESSION_READY_FAILED — Le SDK GameLift Amazon Server n'a pas pu contacter le service pour signaler que la session de jeu était prête.
- INITIALIZATION_MISMATCH — Une méthode client a été appelée après Server : :Initialize (), ou vice versa.
- NOT_INITIALIZED — Le GameLift serveur ou le client Amazon n'a pas été initialisé avec Initialize ().
- NO_TARGET_ALIASID_SET — Aucun AliasID cible n'a été défini.
- NO_TARGET_FLEET_SET — Aucune flotte cible n'a été définie.
- PROCESS_ENDING_FAILED — Le SDK GameLift Amazon Server n'a pas pu contacter le service pour signaler la fin du processus.
- PROCESS_NOT_ACTIVE — Le processus du serveur n'est pas encore actif, n'est pas lié à un et ne peut ni GameSession accepter ni traiter. PlayerSessions
- PROCESS_NOT_READY — Le processus du serveur n'est pas encore prêt à être activé.
- PROCESS_READY_FAILED — Le SDK Amazon GameLift Server n'a pas pu contacter le service pour signaler que le processus était prêt.

- SDK_VERSION_DETECTION_FAILED — La détection de la version du SDK a échoué.
- STX_CALL_FAILED — Un appel au composant principal du serveur XSTx a échoué.
- STX_INITIALIZATION_FAILED — Le composant principal du serveur XSTx n'a pas pu être initialisé.
- UNEXPECTED_PLAYER_SESSION — Le serveur a détecté une session de joueur non enregistrée.
- DÉFAILLANCE DE CONNEXION AU SOCKET WEB
- WEBSOCKET_CONNECT_FAILURE_FORBIDDEN
- WEBSOCKET_CONNECT_FAILURE_INVALID_URL
- WEBSOCKET_CONNECT_FAILURE_TIMEOUT
- WEBSOCKET_RETRIABLE_SEND_MESSAGE_FAILURE — Échec récupérable lors de l'envoi d'un message au service. GameLift WebSocket
- WEBSOCKET_SEND_MESSAGE_FAILURE — Impossible d'envoyer un message au service. GameLift WebSocket
- MATCH_BACKFILL_REQUEST_VALIDATION — La validation de la demande a échoué.
- PLAYER_SESSION_REQUEST_VALIDATION — La validation de la demande a échoué.

PlayerSessionCreationPolicy

Valeur de chaîne indiquant si la session de jeu accepte ou non de nouveaux joueurs. Les valeurs valides sont les suivantes :

- ACCEPT_ALL — Accepte toutes les sessions de nouveaux joueurs.
- DENY_ALL — Refuse toutes les sessions de nouveaux joueurs.
- NOT_SET — La session de jeu n'est pas configurée pour accepter ou refuser les sessions de nouveaux joueurs.

Référence du SDK 3.x pour serveur Amazon GameLift C++

Vous pouvez utiliser cette référence du SDK 3.x pour serveur Amazon GameLift C++ pour vous aider à préparer votre jeu multijoueur en vue de son utilisation avec Amazon. GameLift Pour plus de détails sur le processus d'intégration, consultezAjoutez Amazon GameLift à votre serveur de jeu.

Rubriques

- Référence GameLift du SDK Amazon Server (C++) : Actions

- Référence GameLift du SDK Amazon Server (C++) : types de données

Référence GameLift du SDK Amazon Server (C++) : Actions

Vous pouvez utiliser cette référence du SDK pour serveurs Amazon GameLift C++ pour vous aider à préparer votre jeu multijoueur en vue de son utilisation avec AmazonGameLift. Pour plus de détails sur le processus d'intégration, consultezAjoutez Amazon GameLift à votre serveur de jeu.

Actions

- AcceptPlayerSession()
- ActivateGameSession()
- DescribePlayerSessions()
- GetGameSessionId()
- GetInstanceCertificate()
- GetSdkVersion()
- GetTerminationTime()
- InitSDK()
- ProcessEnding()
- ProcessReady()
- ProcessReadyAsync()
- RemovePlayerSession()
- StartMatchBackfill()
- StopMatchBackfill()
- TerminateGameSession()
- UpdatePlayerSessionCreationPolicy()
- Détruire ()

AcceptPlayerSession()

Informe le GameLift service Amazon qu'un joueur avec l'identifiant de session de joueur spécifié s'est connecté au processus du serveur et doit être validé. Amazon GameLift vérifie que l'identifiant de session du joueur est valide, c'est-à-dire que l'identifiant du joueur a réservé un emplacement de

joueur dans la session de jeu. Une fois la validation effectuée, GameLift Amazon fait passer le statut de la machine à sous RÉSERVÉ à ACTIVE.

Syntaxe

```
GenericOutcome AcceptPlayerSession(const std::string& playerSessionId);
```

Paramètres

playerSessionId

Identifiant unique émis par le GameLift service Amazon en réponse à un appel à l'action CreatePlayerSessionde l'GameLiftAPI Amazon du AWS SDK. Le client du jeu fait référence à cet identifiant lors de la connexion au processus du serveur.

Type : std : :string

Obligatoire : oui

Valeur renvoyée

Renvoie un résultat générique composé d'un succès ou d'un échec avec un message d'erreur.

Exemple

Cet exemple illustre une fonction pour le traitement d'une demande de connexion, y compris la validation et le rejet des ID de session joueur non valides.

```
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()

Indique au GameLift service Amazon que le processus du serveur a démarré une session de jeu et qu'il est désormais prêt à recevoir les connexions des joueurs. Cette action doit être appelée dans le cadre de la fonction de rappel `onStartGameSession()`, une fois que l'initialisation de toutes les sessions de jeu est terminée.

Syntaxe

```
GenericOutcome ActivateGameSession();
```

Paramètres

Cette action n'a aucun paramètre.

Valeur renvoyée

Renvoie un résultat générique composé d'un succès ou d'un échec avec un message d'erreur.

Exemple

Cet exemple illustre l'appel de `ActivateGameSession()` dans le cadre de la fonction de rappel `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()

Récupère les données de session de joueur, y compris les paramètres, les métadonnées de session et les données de joueur. Utilisez cette action pour obtenir des informations pour une seule session de joueur, pour toutes les sessions de joueur d'une session de jeu ou pour toutes les sessions de joueur associées à un seul ID de joueur.

Syntaxe

```
DescribePlayerSessionsOutcome DescribePlayerSessions (
    const Aws::GameLift::Server::Model::DescribePlayerSessionsRequest
 &describePlayerSessionsRequest);
```

Paramètres

describePlayerSessionsRequête

Objet DescribePlayerSessionsRequest décrivant les sessions de joueur à récupérer.

Obligatoire : oui

Valeur renvoyée

En cas de réussite, renvoie un objet `DescribePlayerSessionsOutcome` qui contient un ensemble d'objets de session de joueur correspondant aux paramètres de la demande. Les objets de session du joueur ont une structure identique au type de PlayerSessiondonnées de l'GameLiftAPI Amazon du AWS SDK.

Exemple

Cet exemple illustre une demande de toutes les sessions de joueur activement connectées à une session de jeu spécifiée. En omettant la `Limit` valeur `NextToken` et en la définissant sur 10, Amazon GameLift renvoie les 10 premiers enregistrements des sessions des joueurs correspondant à la demande.

```
// 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()

Extrait un identifiant unique pour la session de jeu actuellement hébergée par le processus serveur, si ce dernier est actif. L'identifiant est retourné dans un format ARN : `arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string>`.

Pour les processus inactifs qui ne sont pas encore activés lors d'une session de jeu, l'appel renvoie `Success` `GameSessionId` = `True` et = `""` (une chaîne vide).

Syntaxe

```
AwsStringOutcome GetGameSessionId();
```

Paramètres

Cette action n'a aucun paramètre.

Valeur renvoyée

En cas de réussite, renvoie l'ID de session de jeu en tant qu'objet `AwsStringOutcome`. En cas d'échec, renvoie un message d'erreur.

Exemple

```
Aws::GameLift::AwsStringOutcome sessionIdOutcome =
    Aws::GameLift::Server::GetGameSessionId();
```

GetInstanceCertificate()

Récupère l'emplacement du fichier d'un certificat TLS codé PEM associé au parc et à ses instances. AWS Certificate Managergénère ce certificat lorsque vous créez une nouvelle flotte avec la configuration du certificat définie sur GENERATED. Utilisez ce certificat pour établir une connexion sécurisée avec un client de jeu et pour chiffrer la communication client/serveur.

Syntaxe

```
GetInstanceCertificateOutcome GetInstanceCertificate();
```

Paramètres

Cette action n'a aucun paramètre.

Valeur renvoyée

En cas de succès, renvoie un `GetInstanceCertificateOutcome` objet contenant l'emplacement du fichier de certificat TLS et de la chaîne de certificats du parc, qui sont stockés sur l'instance. Un fichier de certificat racine, extrait de la chaîne de certificats, est également stocké sur l'instance. En cas d'échec, renvoie un message d'erreur.

Pour plus d'informations sur le certificat et les données de la chaîne de certificats, consultez la section Éléments de GetCertificate réponse dans la référence de l'AWS Certificate ManagerAPI.

Exemple

```
Aws::GameLift::GetInstanceCertificateOutcome certificateOutcome =
    Aws::GameLift::Server::GetInstanceCertificate();
```

GetSdkVersion()

Renvoie numéro de version actuel du kit SDK en cours d'utilisation.

Syntaxe

```
AwsStringOutcome GetSdkVersion();
```

Paramètres

Cette action n'a aucun paramètre.

Valeur renvoyée

En cas de réussite, renvoie la version actuelle du kit SDK en tant qu'objet `AwsStringOutcome`. La chaîne renvoyée inclut uniquement le numéro de version (par exemple « 3.1.5 »). En cas d'échec, renvoie un message d'erreur.

Exemple

```
Aws::GameLift::AwsStringOutcome SdkVersionOutcome =
    Aws::GameLift::Server::GetSdkVersion();
```

GetTerminationTime()

Renvoie l'heure d'arrêt planifiée pour un processus serveur, si une heure de résiliation est disponible. Un processus serveur effectue cette action après avoir reçu un `onProcessTerminate()` rappel du GameLift service Amazon. Amazon GameLift peut appeler `onProcessTerminate()` pour les raisons suivantes : (1) lorsque le processus du serveur a signalé un mauvais fonctionnement ou n'a pas répondu à AmazonGameLift, (2) lors de la résiliation de l'instance lors d'un événement de réduction de la taille ou (3) lorsqu'une instance est résiliée en raison d'une interruption Spot.

Si le processus a reçu un `onProcessTerminate()` rappel, la valeur renvoyée est l'heure de fin estimée. Si le processus n'a pas reçu de `onProcessTerminate()` rappel, un message d'erreur est renvoyé. En savoir plus sur l'arrêt d'un processus serveur.

Syntaxe

```
AwsLongOutcome GetTerminationTime();
```

Paramètres

Cette action n'a aucun paramètre.

Valeur renvoyée

En cas de succès, renvoie l'heure de fin sous forme d'`AwsLongOutcome`objet. La valeur est l'heure de fin, exprimée en ticks écoulés depuis 0001 00:00:00. Par exemple, la valeur date/heure 2020-09-13 12:26:40 -000Z est égale à 637355968000000000 ticks. Si aucune heure de fin n'est disponible, renvoie un message d'erreur.

Exemple

```
Aws::GameLift::AwsLongOutcome TermTimeOutcome =
    Aws::GameLift::Server::GetTerminationTime();
```

InitSDK()

Initialise le GameLift SDK Amazon. Cette méthode doit être appelée au lancement, avant toute autre initialisation GameLift liée à Amazon.

Syntaxe

```
InitSDKOutcome InitSDK();
```

Paramètres

Cette action n'a aucun paramètre.

Valeur renvoyée

En cas de succès, renvoie un InitSdkOutcome objet indiquant que le processus serveur est prêt à être appeléProcessReady().

Exemple

```
Aws::GameLift::Server::InitSDKOutcome initOutcome =
```

```
    Aws::GameLift::Server::InitSDK();
```

ProcessEnding()

Informe le GameLift service Amazon que le processus du serveur est en train de s'arrêter. Cette méthode doit être appelée après toutes les autres tâches de nettoyage, y compris l'arrêt de toutes les sessions de jeu actives. Cette méthode doit quitter avec un code de sortie 0 ; un code de sortie différent de 0 génère un message d'événement indiquant que le processus ne s'est pas fermé correctement.

Une fois que la méthode se termine avec un code de 0, vous pouvez terminer le processus avec un code de sortie réussi. Vous pouvez également quitter le processus avec un code d'erreur. Si vous quittez le site avec un code d'erreur, l'événement du parc indiquera que le processus s'est terminé de manière anormale (`SERVER_PROCESS_TERMINATED_UNHEALTHY`).

Syntaxe

```
GenericOutcome ProcessEnding();
```

Paramètres

Cette action n'a aucun paramètre.

Valeur renvoyée

Renvoie un résultat générique composé d'un succès ou d'un échec avec un message d'erreur.

Exemple

```
Aws::GameLift::GenericOutcome outcome = Aws::GameLift::Server::ProcessEnding();
if (outcome.Success)
    exit(0);  // exit with success
// otherwise, exit with error code
exit(errorCode);
```

ProcessReady()

Indique au GameLift service Amazon que le processus du serveur est prêt à héberger des sessions de jeu. Appelez cette méthode après avoir invoqué InitSDK() et terminé avec succès les tâches de

configuration requises pour que le processus serveur puisse héberger une session de jeu. Cette méthode ne doit être appelée qu'une seule fois par processus.

Cet appel est synchrone. Pour effectuer un appel asynchrone, utilisez ProcessReadyAsync(). Pour plus d'informations, consultez Initialiser le processus du serveur.

Syntaxe

```
GenericOutcome ProcessReady(
    const Aws::GameLift::Server::ProcessParameters &processParameters);
```

Paramètres

processParameters

Objet ProcessParameters communiquant les informations suivantes relatives au processus serveur :

- Noms des méthodes de rappel, implémentées dans le code du serveur de jeu, que le GameLift service Amazon invoque pour communiquer avec le processus du serveur.
- Numéro de port sur lequel le processus serveur écoute.
- Chemin d'accès à tous les fichiers spécifiques à une session de jeu que vous souhaitez qu'Amazon capture GameLift et stocke.

Obligatoire : oui

Valeur renvoyée

Renvoie un résultat générique composé d'un succès ou d'un échec avec un message d'erreur.

Exemple

Cet exemple illustre les implémentations de la fonction d'appel et de rappel ProcessReady().

```
// Set parameters and call ProcessReady
std::string serverLog("serverOut.log");        // Example of a log file written by the
 game server
std::vector<std::string> logPaths;
logPaths.push_back(serverLog);
```

```
int listenPort = 9339;

Aws::GameLift::Server::ProcessParameters processReadyParameter =
 Aws::GameLift::Server::ProcessParameters(
    std::bind(&Server::onStartGameSession, this, std::placeholders::_1),
    std::bind(&Server::onProcessTerminate, this),
    std::bind(&Server::OnHealthCheck, this),
    std::bind(&Server::OnUpdateGameSession, this),
    listenPort,
    Aws::GameLift::Server::LogParameters(logPaths));

Aws::GameLift::GenericOutcome outcome =
   Aws::GameLift::Server::ProcessReady(processReadyParameter);

// Implement callback functions
void Server::onStartGameSession(Aws::GameLift::Model::GameSession myGameSession)
{
   // game-specific tasks when starting a new game session, such as loading map
   GenericOutcome outcome =
       Aws::GameLift::Server::ActivateGameSession (maxPlayers);
}

void Server::onProcessTerminate()
{
   // game-specific tasks required to gracefully shut down a game session,
   // such as notifying players, preserving game state data, and other cleanup
   GenericOutcome outcome = Aws::GameLift::Server::ProcessEnding();
}

bool Server::onHealthCheck()
{
    bool health;
    // complete health evaluation within 60 seconds and set health
    return health;
}
```

ProcessReadyAsync()

Indique au GameLift service Amazon que le processus du serveur est prêt à héberger des sessions de jeu. Cette méthode doit être appelée une fois que le processus serveur est prêt à héberger une session de jeu. Les paramètres spécifient les noms des fonctions de rappel qu'Amazon GameLift peut appeler dans certaines circonstances. Le code du serveur de jeux doit implémenter ces fonctions.

Cet appel est asynchrone. Pour effectuer un appel synchrone, utilisez ProcessReady(). Pour plus d'informations, consultez Initialiser le processus du serveur.

Syntaxe

```
GenericOutcomeCallable ProcessReadyAsync(
    const Aws::GameLift::Server::ProcessParameters &processParameters);
```

Paramètres

processParameters

Objet ProcessParameters communiquant les informations suivantes relatives au processus serveur :

- Noms des méthodes de rappel, implémentées dans le code du serveur de jeu, que le GameLift service Amazon invoque pour communiquer avec le processus du serveur.
- Numéro de port sur lequel le processus serveur écoute.
- Chemin d'accès à tous les fichiers spécifiques à une session de jeu que vous souhaitez qu'Amazon capture GameLift et stocke.

Obligatoire : oui

Valeur renvoyée

Renvoie un résultat générique composé d'un succès ou d'un échec avec un message d'erreur.

Exemple

```
// 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()

Informe le GameLift service Amazon qu'un joueur avec l'identifiant de session de joueur spécifié s'est déconnecté du processus serveur. En réponse, Amazon GameLift change le créneau de joueur pour qu'il soit disponible, ce qui permet de l'attribuer à un nouveau joueur.

Syntaxe

```
GenericOutcome RemovePlayerSession(
    const std::string& playerSessionId);
```

Paramètres

playerSessionId

Identifiant unique émis par le GameLift service Amazon en réponse à un appel à l'action CreatePlayerSessionde l'GameLiftAPI Amazon du AWS SDK. Le client du jeu fait référence à cet identifiant lors de la connexion au processus du serveur.

Type : std : :string

Obligatoire : oui

Valeur renvoyée

Renvoie un résultat générique composé d'un succès ou d'un échec avec un message d'erreur.

Exemple

```
Aws::GameLift::GenericOutcome disconnectOutcome =
    Aws::GameLift::Server::RemovePlayerSession(playerSessionId);
```

StartMatchBackfill()

Envoie une demande de recherche de nouveaux joueurs pour des emplacements ouverts dans une session de jeu créée avec FlexMatch. Voir également l'action du AWS SDK StartMatchBackfill(). Avec cette action, les requêtes de renvoi de correspondance peuvent être initiées par processus de serveur de jeu qui héberge la session de jeu. En savoir plus sur la fonction de FlexMatch remblayage.

Cette action est asynchrone. Si de nouveaux joueurs sont jumelés avec succès, le GameLift service Amazon fournit des données de matchmaking mises à jour en invoquant la fonction de rappel. `OnUpdateGameSession()`

Un processus de serveur ne peut comporter qu'une seule requête de renvoi de correspondance à la fois. Pour envoyer une nouvelle requête, appelez d'abord StopMatchBackfill() pour annuler la requête d'origine.

Syntaxe

```
StartMatchBackfillOutcome StartMatchBackfill (
```

```
    const Aws::GameLift::Server::Model::StartMatchBackfillRequest
 &startBackfillRequest);
```

Paramètres

StartMatchBackfillRequest

Objet StartMatchBackfillRequest qui communique les informations suivantes :

- ID de ticket à attribuer à la requête de renvoi. Ces informations sont facultatives ; si aucun identifiant n'est fourni, Amazon en GameLift générera un automatiquement.
- Matchmaker auquel envoyer la requête. L'ARN de configuration complet est obligatoire. Cette valeur peut être acquise à partir des données matchmaker de la session de jeu.
- ID de la session de jeu en cours de renvoi.
- Données de correspondance disponibles pour les joueurs actuels de la session de jeu.

Obligatoire : oui

Valeur renvoyée

Renvoie un StartMatchBackfillOutcome objet contenant le ticket de remplissage correspondant ou un échec avec un message d'erreur. L'état du ticket peut être suivi à l'aide de l'action du AWS SDK DescribeMatchmaking().

Exemple

```
// 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()

Annule une requête de renvoi de correspondance active qui a été créée avec StartMatchBackfill(). Voir également l'action du AWS SDK StopMatchmaking(). En savoir plus sur la fonction de FlexMatch remblayage.

Syntaxe

```
GenericOutcome StopMatchBackfill (
    const Aws::GameLift::Server::Model::StopMatchBackfillRequest &stopBackfillRequest);
```

Paramètres

StopMatchBackfillRequest

Objet StopMatchBackfillRequest qui identifie le ticket de correspondance à annuler :

- Identifiant de ticket attribué à la requête de renvoi en cours d'annulation
- matchmaker auquel a été envoyée la requête
- session de jeu associée à la requête de renvoi

Obligatoire : oui

Valeur renvoyée

Renvoie un résultat générique composé d'un succès ou d'un échec avec un message d'erreur.

Exemple

```
// 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()

Cette méthode est obsolète depuis la version 4.0.1. Au lieu de cela, le processus du serveur doit appeler ProcessEnding() après la fin d'une session de jeu.

Informe le GameLift service Amazon que le processus du serveur a mis fin à la session de jeu en cours. Cette action est appelée lorsque le processus du serveur reste actif et prêt à héberger une nouvelle session de jeu. Il ne doit être appelé qu'une fois la procédure de fin de session de jeu terminée, car il indique à Amazon GameLift que le processus du serveur est immédiatement disponible pour héberger une nouvelle session de jeu.

Cette action n'est pas appelée si le processus du serveur doit être arrêté après la fin de la session de jeu. Appelez plutôt ProcessEnding() pour signaler que la session de jeu et le processus du serveur se terminent.

Syntaxe

```
GenericOutcome TerminateGameSession();
```

Paramètres

Cette action n'a aucun paramètre.

Valeur renvoyée

Renvoie un résultat générique composé d'un succès ou d'un échec avec un message d'erreur.

UpdatePlayerSessionCreationPolicy()

Met à jour la capacité de la session de jeu à accepter de nouvelles sessions de joueur. Une session de jeu peut être définie pour accepter ou refuser toutes les nouvelles sessions joueur. Voir également l'action du AWS SDK UpdateGameSession().

Syntaxe

```
GenericOutcome UpdatePlayerSessionCreationPolicy(
    Aws::GameLift::Model::PlayerSessionCreationPolicy newPlayerSessionPolicy);
```

Paramètres

newPlayerSessionPolitique

Valeur de chaîne indiquant si la session de jeu accepte ou non de nouveaux joueurs.

Type : Aws : GameLift : :Model : : PlayerSessionCreationPolicy enum. Les valeurs valides sont les suivantes :

- ACCEPT_ALL — Accepte toutes les sessions des nouveaux joueurs.
- DENY_ALL — Refuser toutes les sessions des nouveaux joueurs.

Obligatoire : oui

Valeur renvoyée

Renvoie un résultat générique composé d'un succès ou d'un échec avec un message d'erreur.

Exemple

Cet exemple définit la stratégie de participation de la session de jeu actuelle de manière à ce que tous les joueurs soient acceptés.

```
Aws::GameLift::GenericOutcome outcome =
 Aws::GameLift::Server::UpdatePlayerSessionCreationPolicy(Aws::GameLift::Model::PlayerSessionCr
```

Détruire ()

Nettoie la mémoire allouée par initSDK () lors de l'initialisation du serveur de jeu. Utilisez cette méthode après avoir terminé un processus sur un serveur de jeu pour éviter de gaspiller la mémoire du serveur.

Syntaxe

```
GenericOutcome Aws::GameLift::Server::Destroy();
```

Paramètres

Il n'y a aucun paramètre.

Valeur renvoyée

Renvoie un résultat générique composé d'un succès ou d'un échec avec un message d'erreur.

Exemple

Cet exemple nettoie la mémoire allouée par InitSDK après la fin d'un processus sur un serveur de jeu.

```
if (Aws::GameLift::Server::ProcessEnding().IsSuccess()) {
  Aws::GameLift::Server::Destroy();
  exit(0);
}
```

Référence GameLift du SDK Amazon Server (C++) : types de données

Vous pouvez utiliser cette référence du SDK pour serveurs Amazon GameLift C++ pour vous aider à préparer votre jeu multijoueur en vue de son utilisation avec AmazonGameLift. Pour plus de détails sur le processus d'intégration, consultezAjoutez Amazon GameLift à votre serveur de jeu.

Cette API est définie dans `GameLiftServerAPI.h`, `LogParameters.h` et `ProcessParameters.h`.

- Actions
- Types de données

DescribePlayerSessionsRequest

Ce type de données est utilisé pour spécifier les sessions de joueur à récupérer. Vous pouvez l'utiliser comme suit :

- Fournissez un PlayerSessionId pour demander une session de joueur spécifique.
- Indiquez un GameSessionId pour demander toutes les sessions des joueurs pendant la session de jeu spécifiée.
- Indiquez un PlayerId pour demander toutes les sessions du joueur spécifié.

Pour les volumes importants de sessions de joueur, utilisez les paramètres de pagination pour récupérer les résultats en tant que blocs séquentiels.

Table des matières

GameSessionId

Identifiant de session de jeu unique. Utilisez ce paramètre pour demander toutes les sessions de joueur pour la session de jeu spécifiée. Le format de l'ID de session de jeu est le suivant : `arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string>`. La valeur de **<**ID string**>** peut être une chaîne d'ID personnalisée ou (si spécifiée lors de la création de la session de jeu) une chaîne générée automatiquement.

Type : chaîne

Obligatoire : non

Limite

Nombre maximum de résultats à renvoyer. Utilisez ce paramètre avec NextTokenpour obtenir des résultats sous la forme d'un ensemble de pages séquentielles. Si un ID de session de joueur est spécifié, ce paramètre est ignoré.

Type : entier

Obligatoire : non

NextToken

Jeton indiquant le début de la prochaine page séquentielle de résultats. Utilisez le jeton qui est renvoyé par un appel précédent à cette action. Pour spécifier le début de l'ensemble de résultats, ne spécifiez aucune valeur. Si un ID de session de joueur est spécifié, ce paramètre est ignoré.

Type : chaîne

Obligatoire : non

PlayerId

Identifiant unique pour un joueur. Les ID de joueur sont définis par le développeur. Consultez Générer des identifiants de joueurs.

Type : chaîne

Obligatoire : non

PlayerSessionId

Identifiant unique d'une session de joueur.

Type : chaîne

Obligatoire : non

PlayerSessionStatusFilter

État de session de joueur pour filtrer les résultats. Les états de session de joueur possibles sont les suivants :

- RESERVED - La demande de session de joueur a été reçue, mais le joueur ne s'est pas encore connecté au processus serveur et/ou n'a pas encore été validé.
- ACTIVE - Le joueur a été validé par le processus serveur et est actuellement connecté.
- COMPLETED - La connexion du joueur a été abandonnée.
- TIMEDOUT - Une demande de session de joueur a été reçue, mais le joueur ne s'est pas connecté et/ou n'a pas été validé avant l'expiration du délai (60 secondes).

Type : chaîne

Obligatoire : non

LogParameters

Ce type de données est utilisé pour identifier les fichiers générés pendant une session de jeu que vous souhaitez qu'Amazon GameLift télécharge et stocke une fois la session de jeu terminée. Ces informations sont communiquées au GameLift service Amazon lors d'un ProcessReady() appel.

Table des matières

logPaths

Chemins de répertoire vers les fichiers journaux du serveur de jeu que vous souhaitez qu'Amazon GameLift stocke pour un accès ultérieur. Ces fichiers sont générés au cours de chaque session de jeu. Les chemins et noms des fichiers sont définis dans votre serveur de jeux et stockés dans le répertoire racine de génération de jeu. Les chemins du journal doivent être absolus. Par

exemple, si la version de génération de votre jeu stocke les journaux de session de jeu suivant un chemin tel que `MyGame\sessionlogs\`, le chemin d'accès aux journaux est `c:\game\MyGame\sessionLogs` (sur une instance Windows) ou `/local/game/MyGame/sessionLogs` (sur une instance Linux).

Type : std:vector<std::string>

Obligatoire : non

ProcessParameters

Ce type de données contient l'ensemble des paramètres envoyés au GameLift service Amazon lors d'un ProcessReady() appel.

Table des matières

port

Numéro de port sur lequel le processus serveur écoute les nouvelles connexions de joueur. La valeur doit être comprise dans la plage de ports configurée pour toutes les flottes déployant cette version de génération du serveur de jeux. Ce numéro de port est inclus dans les objets de session de jeu et de session de joueur, que les sessions de jeu utilisent pour se connecter à un processus serveur.

Type : entier

Obligatoire : oui

logParameters

Objet comportant une liste de chemins de répertoires vers les fichiers journaux de sessions de jeu.

Type : Aws : GameLift : :Serveur : : LogParameters

Obligatoire : non

onStartGameSéance

Nom de la fonction de rappel invoquée par le GameLift service Amazon pour activer une nouvelle session de jeu. Amazon GameLift appelle cette fonction en réponse à la demande du client

CreateGameSession. La fonction de rappel transmet un GameSessionobjet (défini dans la référence de l'API Amazon GameLift Service).

Type: `const std::function<void(Aws::GameLift::Model::GameSession)> onStartGameSession`

Obligatoire : oui

onProcessTerminate

Nom de la fonction de rappel invoquée par le GameLift service Amazon pour forcer l'arrêt du processus serveur. Après avoir appelé cette fonction, Amazon GameLift attend cinq minutes que le processus du serveur s'arrête et répond par un ProcessEnding() appel. Si aucune réponse n'est reçue, le processus serveur est arrêté.

Type: `std::function<void()> onProcessTerminate`

Obligatoire : non

onHealthCheck

Nom de la fonction de rappel invoquée par le GameLift service Amazon pour demander un rapport d'état de santé au processus serveur. Amazon GameLift appelle cette fonction toutes les 60 secondes. Après avoir appelé cette fonction, Amazon GameLift attend une réponse pendant 60 secondes et, si aucune réponse n'est reçue, enregistre le processus du serveur comme étant défectueux.

Type: `std::function<bool()> onHealthCheck`

Obligatoire : non

onUpdateGameSéance

Nom de la fonction de rappel invoquée par le GameLift service Amazon pour transmettre un objet de session de jeu mis à jour au processus du serveur. Amazon GameLift fait appel à cette fonction lorsqu'une demande de remplissage de correspondance a été traitée afin de fournir des données de matchmaking mises à jour. Il transmet un GameSessionobjet, une mise à jour de statut (`updateReason`) et l'identifiant du ticket de remplissage du match.

Type: `std::function<void(Aws::GameLift::Server::Model::UpdateGameSession)> onUpdateGameSession`

Obligatoire : non

StartMatchBackfillRequest

Ce type de données est utilisé pour envoyer une requête de renvoi de correspondance. Les informations sont communiquées au GameLift service Amazon lors d'un StartMatchBackfill() appel.

Table des matières

GameSessionArn

Identifiant de session de jeu unique. L'action d'API GetGameSessionId() renvoie l'identifiant au format ARN.

Type : String

Obligatoire : oui

MatchmakingConfigurationArn

Identifiant unique, sous la forme d'un ARN, pour le matchmaker à utiliser pour cette requête. Pour trouver le matchmaker qui a été utilisé pour créer la session de jeu d'origine, recherchez dans l'objet de session de jeu, dans la propriété de données de matchmaker. En savoir plus sur les données de matchmaking dans Word avec les données de matchmaking.

Type : String

Obligatoire : oui

Joueurs

Ensemble de données représentant tous les joueurs qui sont actuellement dans la session de jeu. Le matchmaker utilise ces informations pour rechercher de nouveaux joueurs qui constituent de bonnes correspondances pour les joueurs actuels. Consultez le guide de référence des GameLift API Amazon pour obtenir une description du format d'objet Player. Pour trouver des attributs de joueur, des identifiants et des affectations d'équipe, recherchez dans l'objet de session de jeu, dans la propriété des données de matchmaker. Si une latence est utilisée par le matchmaker, collectez la latence mise à jour pour la région actuelle et incluez-la dans les données de chaque joueur.

Type : standard : vecteurhttps://docs.aws.amazon.com/gamelift/latest/apireference/API_Player.html <player>

Obligatoire : oui

TicketId

Identifiant unique pour une correspondance ou un ticket de requête de renvoi de correspondance. Si aucune valeur n'est fournie ici, Amazon en GameLift générera une sous la forme d'un UUID. Utilisez cet identifiant pour suivre l'état du ticket de renvoi de correspondance ou annuler la requête si nécessaire.

Type : chaîne

Obligatoire : non

StopMatchBackfillRequest

Ce type de données est utilisé pour annuler une demande de renvoi de correspondance. Les informations sont communiquées au GameLift service Amazon lors d'un StopMatchBackfill() appel.

Table des matières

GameSessionArn

Identifiant de session de jeu unique associé à la requête en cours d'annulation.

Type : String

Obligatoire : oui

MatchmakingConfigurationArn

Identifiant unique du matchmaker auquel cette requête a été envoyée.

Type : String

Obligatoire : oui

TicketId

Identifiant unique d'un ticket de requête de correspondance à annuler.

Type : String

Obligatoire : oui

Référence GameLift du SDK Amazon Server pour C#

Vous pouvez utiliser cette référence du SDK pour serveur Amazon GameLift C# pour vous aider à préparer votre jeu multijoueur en vue de son utilisation avec Amazon. GameLift Pour plus de détails sur le processus d'intégration, consultezAjoutez Amazon GameLift à votre serveur de jeu.

Rubriques

- Référence GameLift du SDK Amazon Server 5.x pour C# et Unity
- Référence GameLift du SDK Amazon Server 4.x pour C#

Référence GameLift du SDK Amazon Server 5.x pour C# et Unity

Vous pouvez utiliser cette référence du SDK pour serveurs Amazon GameLift C# 5.x pour vous aider à préparer votre jeu multijoueur en vue de son utilisation avec Amazon. GameLift Pour plus de détails sur le processus d'intégration, voir Ajoutez Amazon GameLift à votre serveur de jeu et pour plus d'informations sur l'utilisation du plug-in SDK du serveur C# pour Unity, voir. Intégrer Amazon GameLift dans un projet Unity Le SDK 5.x GameLift du serveur Amazon pour C# est compatible avec .NET 4.6 et .NET 6.

Rubriques

- Référence GameLift du SDK Amazon Server pour C# et Unity : Actions
- Référence GameLift du SDK Amazon Server pour C# et Unity : types de données

Référence GameLift du SDK Amazon Server pour C# et Unity : Actions

Cette référence du SDK du serveur Amazon GameLift C# vous aide à préparer votre jeu multijoueur pour une utilisation avec Amazon. GameLift Pour plus de détails sur le processus d'intégration, voir Ajoutez Amazon GameLift à votre serveur de jeu et pour plus d'informations sur l'utilisation du plug-in SDK du serveur C# pour Unity, voir. Intégrer Amazon GameLift dans un projet Unity

Actions

- GetSdkVersion()
- InitSDK()
- InitSDK()
- ProcessReady()
- ProcessEnding()

- ActivateGameSession()
- UpdatePlayerSessionCreationPolicy()
- GetGameSessionId()
- GetTerminationTime()
- AcceptPlayerSession()
- RemovePlayerSession()
- DescribePlayerSessions()
- StartMatchBackfill()
- StopMatchBackfill()
- GetComputeCertificate()
- GetFleetRoleCredentials()
- Détruire ()

GetSdkVersion()

Renvoie le numéro de version actuel du kit SDK intégré dans le processus serveur.

Syntaxe

```
AwsStringOutcome GetSdkVersion();
```

Valeur renvoyée

En cas de réussite, renvoie la version actuelle du kit SDK en tant qu'objet the section called "AwsStringOutcome". La chaîne renvoyée inclut le numéro de version (exemple`5.0.0`). En cas d'échec, renvoie un message d'erreur.

Exemple

```
var getSdkVersionOutcome = GameLiftServerAPI.GetSdkVersion();
```

InitSDK()

Initialise le GameLift SDK Amazon pour un parc EC2 géré. Appelez cette méthode au lancement, avant toute autre initialisation liée à Amazon GameLift . Cette méthode lit les paramètres du serveur

depuis l'environnement hôte afin de configurer la communication entre le serveur et le GameLift service Amazon.

Syntaxe

```
GenericOutcome InitSDK();
```

Valeur renvoyée

En cas de succès, renvoie un InitSdkOutcome objet indiquant que le processus serveur est prêt à être appeléProcessReady().

Exemple

```
//Call InitSDK to establish a local connection with the GameLift agent to enable
 further communication.
GenericOutcome initSDKOutcome = GameLiftServerAPI.InitSDK();
```

InitSDK()

Initialise le GameLift SDK Amazon pour une Anywhere flotte. Appelez cette méthode au lancement, avant toute autre initialisation liée à Amazon GameLift . Cette méthode nécessite des paramètres de serveur explicites pour configurer la communication entre le serveur et le GameLift service Amazon.

Syntaxe

```
GenericOutcome InitSDK(ServerParameters serverParameters);
```

Paramètres

ServerParameters

Pour initialiser un serveur de jeu sur une GameLift Anywhere flotte Amazon, créez un `ServerParameters` objet avec les informations suivantes :

- URL WebSocket utilisée pour vous connecter à votre serveur de jeu.
- ID du processus utilisé pour héberger votre serveur de jeu.
- L'ID de l'ordinateur hébergeant les processus de votre serveur de jeu.
- L'ID de la GameLift flotte Amazon contenant votre GameLift Anywhere ordinateur Amazon.
- Le jeton d'autorisation généré par l' GameLift opération Amazon.

Valeur renvoyée

En cas de succès, renvoie un InitSdkOutcome objet indiquant que le processus serveur est prêt à être appeléProcessReady().

> **Note**
>
> Si les appels à échouent pour `InitSDK()` les builds de jeu déployés sur des flottes Anywhere, vérifiez le `ServerSdkVersion` paramètre utilisé lors de la création de la ressource de build. Vous devez définir explicitement cette valeur en fonction de la version du SDK du serveur utilisée. La valeur par défaut de ce paramètre est 4.x, ce qui n'est pas compatible. Pour résoudre ce problème, créez une nouvelle version et déployez-la sur une nouvelle flotte.

Exemple

```
//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()

Indique à Amazon GameLift que le processus du serveur est prêt à héberger des sessions de jeu. Appelez cette méthode après l'avoir invoquée. InitSDK() Cette méthode ne doit être appelée qu'une seule fois par processus.

Syntaxe

```
GenericOutcome ProcessReady(ProcessParameters processParameters)
```

Paramètres

ProcessParameters

Un `ProcessParameters` objet contient des informations sur le processus du serveur.

Valeur renvoyée

Renvoie un résultat générique consistant en un succès ou un échec avec un message d'erreur.

Exemple

Cet exemple illustre à la fois les implémentations de méthodes et de fonctions déléguées.

```
// 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()

Indique à Amazon GameLift que le processus du serveur est en train de se terminer. Appelez cette méthode après toutes les autres tâches de nettoyage (y compris la fermeture de la session de jeu active) et avant de terminer le processus. En fonction du résultat de`ProcessEnding()`, le processus se termine avec succès (0) ou erreur (-1) et génère un événement de flotte. Si le processus se termine par une erreur, l'événement de flotte généré est`SERVER_PROCESS_TERMINATED_UNHEALTHY`.

Syntaxe

```
GenericOutcome ProcessEnding()
```

Valeur renvoyée

Renvoie un résultat générique consistant en un succès ou un échec avec un message d'erreur.

Exemple

Cet exemple appelle `ProcessEnding()` et `Destroy()` avant de terminer le processus du serveur avec un code de sortie de réussite ou d'erreur.

```
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()

Informe Amazon GameLift que le processus du serveur a activé une session de jeu et qu'il est désormais prêt à recevoir les connexions des joueurs. Cette action doit être appelée dans le cadre de la fonction de `onStartGameSession()` rappel, après toute initialisation de session de jeu.

Syntaxe

```
GenericOutcome ActivateGameSession()
```

Valeur renvoyée

Renvoie un résultat générique consistant en un succès ou un échec avec un message d'erreur.

Exemple

Cet exemple illustre l'appel de `ActivateGameSession()` dans le cadre de la fonction déléguée `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()

Met à jour la capacité de la session de jeu à accepter de nouvelles sessions de joueur. Une session de jeu peut être définie pour accepter ou refuser toutes les nouvelles sessions joueur.

Syntaxe

```
GenericOutcome UpdatePlayerSessionCreationPolicy(PlayerSessionCreationPolicy
 playerSessionPolicy)
```

Paramètres

playerSessionPolicy

Valeur de chaîne indiquant si la session de jeu accepte de nouveaux joueurs.

Les valeurs valides sont les suivantes :

- ACCEPT_ALL — Accepte toutes les sessions de nouveaux joueurs.
- DENY_ALL — Refuse toutes les sessions de nouveaux joueurs.

Valeur renvoyée

Renvoie un résultat générique consistant en un succès ou un échec avec un message d'erreur.

Exemple

Cet exemple définit la stratégie de participation de la session de jeu actuelle de manière à ce que tous les joueurs soient acceptés.

```
GenericOutcome updatePlayerSessionPolicyOutcome =

 GameLiftServerAPI.UpdatePlayerSessionCreationPolicy(PlayerSessionCreationPolicy.ACCEPT_ALL);
```

GetGameSessionId()

Récupère l'ID de la session de jeu hébergée par le processus serveur actif.

Pour les processus inactifs qui ne sont pas activés lors d'une session de jeu, l'appel renvoie unthe section called “GameLiftError”.

Syntaxe

```
AwsStringOutcome GetGameSessionId()
```

Valeur renvoyée

En cas de réussite, renvoie l'ID de session de jeu en tant qu'objet the section called “AwsStringOutcome”. En cas d'échec, renvoie un message d'erreur. »

Exemple

```
AwsStringOutcome getGameSessionIdOutcome = GameLiftServerAPI.GetGameSessionId();
```

GetTerminationTime()

Renvoie l'heure d'arrêt planifiée pour un processus serveur, si une heure de résiliation est disponible. Un processus serveur exécute cette action après avoir reçu un `onProcessTerminate()` rappel d'Amazon GameLift. Amazon GameLift appelle `onProcessTerminate()` pour les raisons suivantes :

- Lorsque le processus du serveur a signalé un mauvais état de santé ou n'a pas répondu à Amazon GameLift.
- Lorsque vous mettez fin à l'instance lors d'un événement de réduction de la taille.
- Lorsqu'une instance est interrompue en raison d'une interruption ponctuelle.

Syntaxe

```
AwsDateTimeOutcome GetTerminationTime()
```

Valeur renvoyée

En cas de succès, renvoie l'heure de fin sous forme d'the section called “AwsDateTimeOutcome”objet. La valeur est le délai de résiliation, exprimé en ticks écoulés depuis.

`0001 00:00:00` Par exemple, la valeur de la date et de l'heure `2020-09-13 12:26:40 -000Z` est égale à celle `637355968000000000` des ticks. Si aucune heure de résiliation n'est disponible, renvoie un message d'erreur.

Exemple

```
AwsDateTimeOutcome getTerminationTimeOutcome = GameLiftServerAPI.GetTerminationTime();
```

AcceptPlayerSession()

Informe Amazon GameLift qu'un joueur possédant l'identifiant de session de joueur spécifié s'est connecté au processus du serveur et doit être validé. Amazon GameLift vérifie que l'identifiant de session du joueur est valide. Une fois la session du joueur validée, Amazon GameLift change le statut de l'emplacement du joueur de RÉSERVÉ à ACTIF.

Syntaxe

```
GenericOutcome AcceptPlayerSession(String playerSessionId)
```

Paramètres

playerSessionId

Identifiant unique émis GameLift lors de la création d'une nouvelle session de joueur.

Valeur renvoyée

Renvoie un résultat générique consistant en un succès ou un échec avec un message d'erreur.

Exemple

Cet exemple illustre une fonction pour le traitement d'une demande de connexion, y compris la validation et le rejet des ID de session joueur non valides.

```
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()

Indique à Amazon GameLift qu'un joueur s'est déconnecté du processus du serveur. En réponse, Amazon GameLift modifie l'emplacement du joueur pour le rendre disponible.

Syntaxe

```
GenericOutcome RemovePlayerSession(String playerSessionId)
```

Paramètres

playerSessionId

ID unique émis par Amazon GameLift lors de la création d'une nouvelle session de joueur.

Valeur renvoyée

Renvoie un résultat générique consistant en un succès ou un échec avec un message d'erreur.

Exemple

```
GenericOutcome removePlayerSessionOutcome =
 GameLiftServerAPI.RemovePlayerSession(playerSessionId);
```

DescribePlayerSessions()

Récupère les données de session du joueur, notamment les paramètres, les métadonnées de session et les données du joueur. Utilisez cette action pour obtenir des informations pour une seule session de joueur, pour toutes les sessions de joueur d'une session de jeu ou pour toutes les sessions de joueur associées à un seul ID de joueur.

Syntaxe

```
DescribePlayerSessionsOutcome DescribePlayerSessions(DescribePlayerSessionsRequest
 describePlayerSessionsRequest)
```

Paramètres

DescribePlayerSessionsRequest

the section called “DescribePlayerSessionsRequest”Objet qui décrit les sessions de joueur à récupérer.

Valeur renvoyée

En cas de succès, renvoie un the section called “DescribePlayerSessionsOutcome” objet contenant un ensemble d'objets de session de joueur correspondant aux paramètres de la demande.

Exemple

Cet exemple illustre une demande de toutes les sessions de joueur activement connectées à une session de jeu spécifiée. En omettant NextTokenet en fixant la valeur limite à 10, Amazon GameLift renverra les 10 premiers enregistrements de session de joueur correspondant à la demande.

```
// 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()

Envoie une demande de recherche de nouveaux joueurs pour des emplacements ouverts dans une session de jeu créée avec FlexMatch. Pour plus d'informations, voir la fonction de FlexMatch remblayage.

Cette action est asynchrone. Si de nouveaux joueurs sont jumelés, Amazon GameLift fournit des données de matchmaking mises à jour à l'aide de la fonction de rappel. `OnUpdateGameSession()`

Un processus de serveur ne peut comporter qu'une seule requête de renvoi de correspondance à la fois. Pour envoyer une nouvelle requête, appelez d'abord StopMatchBackfill() pour annuler la requête d'origine.

Syntaxe

```
StartMatchBackfillOutcome StartMatchBackfill (StartMatchBackfillRequest
 startBackfillRequest);
```

Paramètres

StartMatchBackfillRequest

Un `StartMatchBackfillRequest` objet contient des informations relatives à la demande de remblayage.

Valeur renvoyée

Renvoie un the section called “StartMatchBackfillOutcome” objet avec l'identifiant du ticket de remplacement correspondant, ou un échec avec un message d'erreur.

Exemple

```
// 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()

Annule une demande de remplacement de match active. Pour plus d'informations, voir la fonction de FlexMatch remblayage.

Syntaxe

```
GenericOutcome StopMatchBackfill (StopMatchBackfillRequest stopBackfillRequest);
```

Paramètres

StopMatchBackfillRequest

Un `StopMatchBackfillRequest` objet qui fournit des détails sur le ticket de matchmaking que vous êtes en train d'arrêter.

Valeur renvoyée

Renvoie un résultat générique consistant en un succès ou un échec avec un message d'erreur.

Exemple

```
// 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()

Récupère le chemin d'accès au certificat TLS utilisé pour chiffrer la connexion réseau entre le serveur de jeu et votre client de jeu. Vous pouvez utiliser le chemin du certificat lorsque vous enregistrez votre appareil informatique dans une GameLift Anywhere flotte Amazon. Pour plus d'informations, voir, RegisterCompute.

Syntaxe

```
GetComputeCertificateOutcome GetComputeCertificate();
```

Valeur renvoyée

Renvoie un GetComputeCertificateResponse objet contenant les éléments suivants :

- CertificatePath: chemin d'accès au certificat TLS sur votre ressource de calcul. Lorsque vous utilisez une flotte GameLift gérée par Amazon, ce chemin contient :
  - `certificate.pem`: le certificat d'utilisateur final. La chaîne de certificats complète est la combinaison des `certificateChain.pem` éléments ajoutés à ce certificat.
  - `certificateChain.pem`: chaîne de certificats qui contient le certificat racine et les certificats intermédiaires.
  - `rootCertificate.pem`: le certificat racine.
  - `privateKey.pem`: clé privée pour le certificat d'utilisateur final.
- ComputeName: nom de votre ressource de calcul.

Exemple

```
GetComputeCertificateOutcome getComputeCertificateOutcome =
 GameLiftServerAPI.GetComputeCertificate();
```

GetFleetRoleCredentials()

Récupère les informations d'identification du rôle IAM qui autorisent Amazon GameLift à interagir avec d'autres. Services AWS Pour plus d’informations, consultez Communiquez avec les autres AWS ressources de vos flottes.

Syntaxe

```
GetFleetRoleCredentialsOutcome GetFleetRoleCredentials(GetFleetRoleCredentialsRequest
 request);
```

Paramètres

GetFleetRoleCredentialsRequest

Des informations d'identification de rôle qui étendent un accès limité à vos AWS ressources au serveur de jeu.

Valeur renvoyée

Renvoie un objet the section called “GetFleetRoleCredentialsOutcome”.

Exemple

```
// form the fleet credentials request
GetFleetRoleCredentialsRequest getFleetRoleCredentialsRequest = new
 GetFleetRoleCredentialsRequest(){
  RoleArn = "arn:aws:iam::123456789012:role/service-role/exampleGameLiftAction"
};
GetFleetRoleCredentialsOutcome GetFleetRoleCredentialsOutcome credentials =
 GetFleetRoleCredentials(getFleetRoleCredentialsRequest);
```

Détruire ()

Libère de la mémoire le SDK du serveur de GameLift jeu Amazon. Il est recommandé d'appeler cette méthode après `ProcessEnding()` et avant de terminer le processus. Si vous utilisez une flotte Anywhere et que vous n'interrompez pas les processus du serveur après chaque session de jeu, appelez `Destroy()` puis réinitialisez avant `InitSDK()` d'informer Amazon GameLift que le processus est prêt à héberger une session de jeu avec. `ProcessReady()`

Syntaxe

```
GenericOutcome Destroy()
```

Valeur renvoyée

Renvoie un résultat générique consistant en un succès ou un échec avec un message d'erreur.

Exemple

```
// 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);
  }
```

Référence GameLift du SDK Amazon Server pour C# et Unity : types de données

Cette référence du SDK Amazon GameLift C# Server peut vous aider à préparer votre jeu multijoueur en vue de son utilisation avec Amazon. GameLift Pour plus de détails sur le processus d'intégration, voir Ajoutez Amazon GameLift à votre serveur de jeu et pour plus d'informations sur l'utilisation du plug-in SDK du serveur C# pour Unity, voir. Intégrer Amazon GameLift dans un projet Unity

Types de données

- LogParameters
- ProcessParameters
- UpdateGameSession
- GameSession
- ServerParameters
- StartMatchBackfillRequest
- Joueur
- DescribePlayerSessionsRequest
- StopMatchBackfillRequest
- GetFleetRoleCredentialsRequest

- AttributeValue
- AwsStringOutcome
- GenericOutcome
- DescribePlayerSessionsOutcome
- DescribePlayerSessionsResult
- PlayerSession
- StartMatchBackfillOutcome
- StartMatchBackfillResult
- GetComputeCertificateOutcome
- GetComputeCertificateResult
- GetFleetRoleCredentialsOutcome
- GetFleetRoleCredentialsResult
- AwsDateTimeOutcome
- GameLiftError
- Enums

LogParameters

Utilisez ce type de données pour identifier les fichiers générés pendant une session de jeu que vous souhaitez que le serveur de jeu charge sur Amazon une GameLift fois la session de jeu terminée. Le serveur de jeu communique avec `LogParameters to` Amazon GameLift lors d'un ProcessReady() appel.

Propriétés	Description
LogPaths	La liste des chemins de répertoire vers les fichiers journaux du serveur de jeu que vous souhaitez GameLift qu'Amazon stocke pour un accès ultérieur. Le processus du serveur génère ces fichiers lors de chaque session de jeu. Vous définissez les chemins et les noms des fichiers sur votre serveur de jeu et vous les stockez dans le répertoire de compilation racine du jeu.

	Les chemins du journal doivent être absolus. Par exemple, si la version de votre jeu stocke les journaux des sessions de jeu dans un chemin tel que `MyGame\sessionLogs\` , alors ce chemin se `c:\game\MyGame\sessionLogs` trouve sur une instance Windows. Type : `List<String>` Obligatoire : non

ProcessParameters

Ce type de données contient l'ensemble des paramètres envoyés à Amazon GameLift lors d'un ProcessReady() appel.

Propriétés	Description
LogParameters	L'objet contenant la liste des chemins de répertoire vers les fichiers journaux des sessions de jeu. Type : `Aws::GameLift::Server::` LogParameters Obligatoire : oui
OnHealthCheck	Nom de la fonction de rappel GameLift invoquée par Amazon pour demander un rapport d'état de santé au processus du serveur. Amazon GameLift appelle cette fonction toutes les 60 secondes. Après avoir appelé cette fonction, Amazon GameLift attend une réponse pendant 60 secondes. Si aucune réponse n'est reçue, Amazon GameLift enregistre le processus du serveur comme étant défectueux.

	Type : `void OnHealthCheckDelegate()` Obligatoire : oui
OnProcessTerminate	Le nom de la fonction de rappel GameLift invoquée par Amazon pour forcer l'arrêt du processus du serveur. Après avoir appelé cette fonction, Amazon GameLift attend cinq minutes que le processus serveur s'arrête et répond par un ProcessEnding() appel avant d'arrêter le processus serveur. Type : `void OnProcessTerminate Delegate()` Obligatoire : oui
OnStartGameSession	Le nom de la fonction de rappel GameLift invoquée par Amazon pour activer une nouvelle session de jeu. Amazon GameLift appelle cette fonction en réponse à la demande du client CreateGameSession. La fonction de rappel prend un GameSessionobjet défini dans la référence des GameLift API Amazon. Type : `void OnStartGameSession Delegate( GameSession )` Obligatoire : oui

OnUpdateGameSession	Nom de la fonction de rappel GameLift invoquée par Amazon pour transmettre un objet de session de jeu mis à jour au processus du serveur. Amazon GameLift fait appel à cette fonction lorsqu'une demande de remplissage de correspondance a été traitée afin de fournir des données de matchmaking mises à jour. Il transmet un GameSessionobjet, une mise à jour de statut (`updateReason`) et l'identifiant du ticket de remplissage du match. Type : vide OnUpdateGameSessionDelegate (UpdateGameSession) Obligatoire : non
Port	Le numéro de port que le processus serveur écoute pour les connexions de nouveaux joueurs. La valeur doit être comprise dans la plage de ports configurée pour toutes les flottes déployant cette version de génération du serveur de jeux. Ce numéro de port est inclus dans les objets de session de jeu et de session de joueur, que les sessions de jeu utilisent pour se connecter à un processus serveur. Type : `Integer` Obligatoire : oui

UpdateGameSession

Les informations mises à jour pour un objet de session de jeu incluent la raison pour laquelle la session de jeu a été mise à jour. Si la mise à jour est liée à une action de remplissage correspondant, ce type de données inclut l'ID du ticket de remplissage.

Propriétés	Description
GameSession	Un GameSessionobjet défini par l'GameLiftAPI Amazon. L'`GameSession` objet contient des propriétés décrivant une session de jeu. Type : `GameSession GameSession()` Obligatoire : oui
UpdateReason	La raison pour laquelle la session de jeu est mise à jour. Type : `UpdateReason UpdateReason()` Obligatoire : oui
BackfillTicketId	L'ID du ticket de remplissage qui tente de mettre à jour la session de jeu. Type : `String` Obligatoire : oui

GameSession

Détails d'une session de jeu.

Propriétés	Description
GameSessionId	Un identifiant unique pour la session de jeu. Un ARN de session de jeu a le format suivant :`arn:aws:gamelift:<region>::gamesession/<fleet ID>/<custom ID string or idempotency token>` . Type : `String` Obligatoire : non

Propriétés	Description
Nom	Une étiquette descriptive de la session de jeu. Type : `String` Obligatoire : non
FleetId	Un identifiant unique pour la flotte sur laquelle se déroule la session de jeu. Type : `String` Obligatoire : non
MaximumPlayerSessionCount	Le nombre maximum de connexions de joueurs à la session de jeu. Type : `Integer` Obligatoire : non
Port	Le numéro de port de la session de jeu. Pour se connecter à un serveur de GameLift jeu Amazon, une application a besoin à la fois de l'adresse IP et du numéro de port. Type : `Integer` Obligatoire : non
IpAddress	Adresse IP de la session de jeu. Pour se connecter à un serveur de GameLift jeu Amazon, une application a besoin à la fois de l'adresse IP et du numéro de port. Type : `String` Obligatoire : non

Propriétés	Description
GameSessionData	Ensemble de propriétés de session de jeu personnalisées, mises en forme en tant que valeur de chaîne unique. Type : `String` Obligatoire : non
MatchmakerData	Les informations sur le processus de matchmaking qui a été utilisé pour créer la session de jeu, en syntaxe JSON, formatées sous forme de chaîne. En plus de la configura tion de matchmaking utilisée, elle contient des données sur tous les joueurs affectés au match, y compris les attributs des joueurs et les affectations des équipes. Type : `String` Obligatoire : non
GameProperties	Ensemble de propriétés personnalisées pour une session de jeu, formatées sous forme de paires clé:valeur. Ces propriétés sont transmise s avec une demande de démarrage d'une nouvelle session de jeu. Type : `Dictionary<string, string>` Obligatoire : non

Propriétés	Description
DnsName	L'identifiant DNS attribué à l'instance qui exécute la session de jeu. Les valeurs ont le format suivant : • Flottes compatibles TLS :. `<unique identifier>.<region identifier>.amazongamelift.com` • Flottes non compatibles TLS :. `ec2-<unique identifier>.compute.amazonaws.com` Lorsque vous vous connectez à une session de jeu exécutée sur une flotte compatible TLS, vous devez utiliser le nom DNS et non l'adresse IP. Type : `String` Obligatoire : non

ServerParameters

Informations utilisées pour maintenir la connexion entre un GameLift Anywhere serveur Amazon et le GameLift service Amazon. Ces informations sont utilisées lors du lancement de nouveaux processus serveur avecInitSDK(). Pour les serveurs hébergés sur des instances EC2 GameLift gérées par Amazon, utilisez un objet vide.

Propriétés	Description
WebSocketUrl	Ils `GameLiftServerSdkEndpoint` sont retournés lorsque vous `RegisterCompute` faisiez partie d'Amazon GameLiftAnywhere. Type : `String`

Propriétés	Description
	Obligatoire : oui
ProcessId	Un identifiant unique enregistré auprès du processus serveur hébergeant votre jeu. Type : `String` Obligatoire : oui
HostId	Un identifiant unique pour l'hôte associé aux processus du serveur hébergeant votre jeu. L'identifiant d'hôte est celui ComputeName utilisé lorsque vous avez enregistré votre ordinateur. Pour plus d'informations, voir, RegisterCompute Type : `String` Obligatoire : oui
FleetId	L'ID du parc dans lequel le calcul est enregistré. Pour plus d'informations, voir RegisterCompute. Type : `String` Obligatoire : oui
AuthToken	Le jeton d'authentification généré par Amazon GameLift qui authentifie votre serveur auprès d'Amazon. GameLift Pour plus d'informations, voir GetComputeAuthToken. Type : `String` Obligatoire : oui

StartMatchBackfillRequest

Informations utilisées pour créer une demande de remplissage de matchmaking. Le serveur de jeu communique ces informations à Amazon GameLift lors d'un StartMatchBackfill() appel.

Propriétés	Description
GameSessionArn	L'identifiant unique de la session de jeu. L'opération d'API `GetGameSessionId` renvoie l'identifiant au format ARN. Type : `String` Obligatoire : oui
MatchmakingConfigurationArn	L'identifiant unique, sous la forme d'un ARN, que le système de matchmaking doit utiliser pour cette demande. L'ARN du matchmaker pour la session de jeu d'origine se trouve dans l'objet de session de jeu de la propriété de données du matchmaker. Pour en savoir plus sur les données de matchmaking, consultez la section Travailler avec les données de matchmaking. Type : `String` Obligatoire : oui
Joueurs	Un ensemble de données qui représente tous les joueurs qui participent actuellement à la session de jeu. Le matchmaker utilise ces informations pour rechercher de nouveaux joueurs qui constituent de bonnes correspon dances pour les joueurs actuels. Type : `List<Player>` Obligatoire : oui

Propriétés	Description
TicketId	L'identifiant unique d'un ticket de matchmaki ng ou de demande de remplissage de match. Si vous ne fournissez aucune valeur, Amazon en GameLift génère une. Utilisez cet identifia nt pour suivre l'état du ticket de renvoi de correspondance ou annuler la requête si nécessaire. Type : `String` Obligatoire : non

Joueur

Représente un joueur dans le matchmaking. Lorsqu'une demande de matchmaking commence, un joueur possède un identifiant de joueur, des attributs et éventuellement des données de latence. Amazon GameLift ajoute des informations sur l'équipe après un match.

Propriétés	Description
LatencyInMS	Ensemble de valeurs exprimées en milliseco ndes, qui indiquent le niveau de latence qu'un joueur ressent lorsqu'il est connecté à un lieu. Si cette propriété est utilisée, le joueur est sélectionné uniquement pour les lieux répertori és. Si un matchmaker dispose d'une règle qui évalue la latence, les joueurs doivent indiquer la latence pour être mis en relation. Type : `Dictionary<string, int>` Obligatoire : non
PlayerAttributes	Une collection de paires clé:valeur qui contiennent des informations sur les joueurs à

Propriétés	Description
	utiliser dans le matchmaking. Les clés d'attribut du joueur doivent correspondre à PlayerAttributes celles utilisées dans un ensemble de règles de matchmaking. Pour plus d'informations sur les attributs des joueurs, consultez AttributeValue. Type : `Dictionary<string, AttributeValue` Obligatoire : non
PlayerId	Un identifiant unique pour un joueur. Type : `String` Obligatoire : non
Equipe	Le nom de l'équipe à laquelle le joueur est affecté lors d'un match. Vous définissez le nom de l'équipe dans le jeu de règles de matchmaking. Type : `String` Obligatoire : non

DescribePlayerSessionsRequest

Ce type de données est utilisé pour spécifier les sessions de joueur à récupérer. Il peut être utilisé de plusieurs manières : (1) PlayerSessionId pour demander une session de joueur spécifique ; (2) GameSessionId pour demander toutes les sessions des joueurs pendant la session de jeu spécifiée ; ou (3) PlayerId pour demander toutes les sessions des joueurs pour le joueur spécifié. Pour les volumes importants de sessions de joueur, utilisez les paramètres de pagination pour récupérer les résultats en tant que pages séquentielles.

Propriétés	Description
GameSessionId	L'identifiant unique de la session de jeu. Utilisez ce paramètre pour demander toutes les sessions de joueur pour la session de jeu spécifiée. Le format de l'ID de session de jeu est le suivant : `arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string>` . La valeur de <ID string>peut être une chaîne d'ID personnalisée (si spécifiée lors de la création de la session de jeu) ou une chaîne générée automatiquement. Type : `String` Obligatoire : non
PlayerSessionId	L'identifiant unique d'une session de joueur. Type : `String` Obligatoire : non
PlayerId	L'identifiant unique d'un joueur. Consultez Générer des identifiants de joueurs. Type : `String` Obligatoire : non
PlayerSessionStatusFilter	État de la session du joueur sur lequel filtrer les résultats. Les états de session de joueur possibles sont les suivants : • RESERVED - La demande de session de joueur a été reçue, mais le joueur ne s'est pas encore connecté au processus serveur et/ou n'a pas encore été validé.

Propriétés	Description
	• ACTIVE - Le joueur a été validé par le processus serveur et est actuellement connecté. • COMPLETED - La connexion du joueur a été abandonnée. • TIMEDOUT - Une demande de session de joueur a été reçue, mais le joueur ne s'est pas connecté et/ou n'a pas été validé avant l'expiration du délai (60 secondes). Type : `String` Obligatoire : non
NextToken	Le jeton indiquant le début de la page de résultats suivante. Pour spécifier le début du jeu de résultats, ne fournissez pas de valeur. Si vous fournissez un identifiant de session de joueur, ce paramètre est ignoré. Type : `String` Obligatoire : non
Limite	Nombre maximal de résultats à renvoyer. Si vous fournissez un identifiant de session de joueur, ce paramètre est ignoré. Type : `int` Obligatoire : non

StopMatchBackfillRequest

Informations utilisées pour annuler une demande de remplissage de matchmaking. Le serveur de jeu communique ces informations au GameLift service Amazon lors d'un StopMatchBackfill() appel.

Propriétés	Description
GameSessionArn	L'identifiant unique de session de jeu de la demande annulée. Type : `string` Obligatoire : oui
MatchmakingConfigurationArn	L'identifiant unique du système de matchmaki ng auquel cette demande a été envoyée. Type : `string` Obligatoire : oui
TicketId	L'identifiant unique du ticket de demande de remplissage à annuler. Type : `string` Obligatoire : oui

GetFleetRoleCredentialsRequest

Ce type de données donne au serveur de jeu un accès limité à vos autres AWS ressources. Pour plus d'informations, consultez Configurer un rôle de service IAM pour Amazon GameLift.

Propriétés	Description
RoleArn	Le nom de ressource Amazon (ARN) du rôle de service qui étend l'accès limité à vos AWS ressources. Type : `string` Obligatoire : oui
RoleSessionName	Le nom de la session qui décrit l'utilisation des informations d'identification du rôle.

Propriétés	Description
	Type : `string` Obligatoire : non

AttributeValue

Utilisez ces valeurs dans des paires Joueur clé-valeur d'attribut. Cet objet vous permet de spécifier une valeur d'attribut à l'aide de n'importe quel type de données valide : chaîne, nombre, tableau de chaînes ou carte de données. Chaque `AttributeValue` objet ne peut utiliser qu'une seule des propriétés disponibles.

Propriétés	Description
AttrType	Spécifie le type de valeur d'attribut. Type : valeur d'`AttrType`énumération. Obligatoire : non
S	Représente la valeur d'un attribut de chaîne. Type : `string` Obligatoire : oui
N	Représente une valeur d'attribut numérique. Type : `double` Obligatoire : oui
SL	Représente un tableau de valeurs d'attributs de chaîne. Type : `string[]` Obligatoire : oui

Propriétés	Description
SDM	Représente un dictionnaire de clés de chaîne et de valeurs doubles. Type : `Dictionary<string, double>` Obligatoire : oui

AwsStringOutcome

Ce type de données résulte d'une action et produit un objet doté des propriétés suivantes :

Propriétés	Description
Résultat	Le résultat de l'action. Type : `string` Obligatoire : non
Réussite	Si l'action a été couronnée de succès ou non. Type : `bool` Obligatoire : oui
Erreur	L'erreur qui s'est produite si l'action a échoué. Type : the section called “GameLiftError” Obligatoire : non

GenericOutcome

Ce type de données résulte d'une action et produit un objet doté des propriétés suivantes :

Propriétés	Description
Réussite	Si l'action a été couronnée de succès ou non.

Propriétés	Description
	Type : `bool` Obligatoire : oui
Erreur	L'erreur qui s'est produite si l'action a échoué. Type : the section called “GameLiftError” Obligatoire : non

DescribePlayerSessionsOutcome

Ce type de données résulte d'une action et produit un objet doté des propriétés suivantes :

Propriétés	Description
Résultat	Le résultat de l'action. Type : the section called “DescribePlayerSessionsResult” Obligatoire : non
Réussite	Si l'action a été couronnée de succès ou non. Type : `bool` Obligatoire : oui
Erreur	L'erreur qui s'est produite si l'action a échoué. Type : the section called “GameLiftError” Obligatoire : non

DescribePlayerSessionsResult

Propriétés	Description
NextToken	Le jeton indiquant le début de la page de résultats suivante. Pour spécifier le début du jeu de résultats, ne fournissez pas de valeur. Si vous fournissez un identifiant de session de joueur, ce paramètre est ignoré. Type : `string` Obligatoire : oui
PlayerSessions	Une collection d'objets contenant des propriété s correspondant à la demande pour chaque session de joueur. Type : `IList<`the section called “PlayerSession” `>` Obligatoire :
Réussite	Si l'action a été couronnée de succès ou non. Type : `bool` Obligatoire : oui
Erreur	L'erreur qui s'est produite si l'action a échoué. Type : the section called “GameLiftError” Obligatoire : non

PlayerSession

Propriétés	Description
CreationTime	Type : `long`

Propriétés	Description
	Obligatoire : oui
FleetId	Type : `string` Obligatoire : oui
GameSessionId	Type : `string` Obligatoire : oui
IpAddress	Type : `string` Obligatoire : oui
PlayerData	Type : `string` Obligatoire : oui
PlayerId	Type : `string` Obligatoire : oui
PlayerSessionId	Type : `string` Obligatoire : oui
Port	Type : `int` Obligatoire : oui
État	Type : Une `PlayerSessionStatus` énumération. Obligatoire : oui
TerminationTime	Type : `long` Obligatoire : oui

Propriétés	Description
DnsName	Type : `string` Obligatoire : oui

StartMatchBackfillOutcome

Ce type de données résulte d'une action et produit un objet doté des propriétés suivantes :

Propriétés	Description
Résultat	Le résultat de l'action. Type : the section called “StartMatchBackfillResult” Obligatoire : non
Réussite	Si l'action a été couronnée de succès ou non. Type : `bool` Obligatoire : oui
Erreur	L'erreur qui s'est produite si l'action a échoué. Type : the section called “GameLiftError” Obligatoire : non

StartMatchBackfillResult

Propriétés	Description
TicketId	Type : `string` Obligatoire : oui

GetComputeCertificateOutcome

Ce type de données résulte d'une action et produit un objet doté des propriétés suivantes :

Propriétés	Description
Résultat	Le résultat de l'action. Type : the section called “GetComputeCertificateResult” Obligatoire : non
Réussite	Si l'action a été couronnée de succès ou non. Type : `bool` Obligatoire : oui
Erreur	L'erreur qui s'est produite si l'action a échoué. Type : the section called “GameLiftError” Obligatoire : non

GetComputeCertificateResult

Le chemin d'accès au certificat TLS sur votre ordinateur et le nom d'hôte de l'ordinateur.

Propriétés	Description
CertificatePath	Type : `string` Obligatoire : oui
ComputeName	Type : `string` Obligatoire : oui

GetFleetRoleCredentialsOutcome

Ce type de données résulte d'une action et produit un objet doté des propriétés suivantes :

Propriétés	Description
Résultat	Le résultat de l'action. Type : the section called “GetFleetRoleCredentialsResult” Obligatoire : non
Réussite	Si l'action a été couronnée de succès ou non. Type : `bool` Obligatoire : oui
Erreur	L'erreur qui s'est produite si l'action a échoué. Type : the section called “GameLiftError” Obligatoire : non

GetFleetRoleCredentialsResult

Propriétés	Description
AccessKeyId	L'ID de clé d'accès permettant d'authentifier et de donner accès à vos AWS ressources. Type : `string` Obligatoire : non
AssumedRoleId	L'ID de l'utilisateur auquel appartient le rôle de service. Type : `string`

Propriétés	Description
	Obligatoire : non
AssumedRoleUserArn	Le nom de ressource Amazon (ARN) de l'utilisateur auquel appartient le rôle de service. Type : `string` Obligatoire : non
Expiration	Le délai avant l'expiration de vos informations d'identification de session. Type : `DateTime` Obligatoire : non
SecretAccessKey	L'ID de clé d'accès secrète pour l'authentification. Type : `string` Obligatoire : non
SessionToken	Un jeton pour identifier la session active en cours qui interagit avec vos AWS ressources. Type : `string` Obligatoire : non
Réussite	Si l'action a été couronnée de succès ou non. Type : `bool` Obligatoire : oui

Propriétés	Description
Erreur	L'erreur qui s'est produite si l'action a échoué. Type : the section called “GameLiftError” Obligatoire : non

AwsDateTimeOutcome

Ce type de données résulte d'une action et produit un objet doté des propriétés suivantes :

Propriétés	Description
Résultat	Le résultat de l'action. Type : `DateTime` Obligatoire : non
Réussite	Si l'action a été couronnée de succès ou non. Type : `bool` Obligatoire : oui
Erreur	L'erreur qui s'est produite si l'action a échoué. Type : the section called “GameLiftError” Obligatoire : non

GameLiftError

Propriétés	Description
ErrorType	Type d'erreur.

Propriétés	Description
	Type : Une `GameLiftErrorType` énumération. Obligatoire : non
ErrorName	Le nom de l'erreur. Type : `string` Obligatoire : non
ErrorMessage	Message d’erreur. Type : `string` Obligatoire : non

Enums

Les énumérations définies pour le SDK GameLift du serveur Amazon (C#) sont définies comme suit :

AttrType

- NONE
- FICELLE
- DOUBLE
- LISTE_CHAÎNE
- STRING_DOUBLE_MAP

GameLiftErrorType

Valeur de chaîne indiquant le type d'erreur. Les valeurs valides sont les suivantes :

- SERVICE_CALL_FAILED — Un appel à un AWS service a échoué.
- LOCAL_CONNECTION_FAILED — La connexion locale à Amazon a échoué. GameLift
- NETWORK_NOT_INITIALIZED — Le réseau n'a pas été initialisé.
- GAMESSION_ID_NOT_SET — L'identifiant de session de jeu n'a pas été défini.
- EXCEPTION DE DEMANDE INCORRECTE

- EXCEPTION DE SERVICE INTERNE
- ALREADY_INITIALIZED — Le GameLift serveur ou le client Amazon a déjà été initialisé avec Initialize ().
- FLEET_MISMATCH — La flotte cible ne correspond pas à la flotte d'une GameSession ou d'une PlayerSession.
- GAMELIFT_CLIENT_NOT_INITIALIZED — Le client Amazon n'a pas été initialisé. GameLift
- GAMELIFT_SERVER_NOT_INITIALIZED — Le serveur Amazon n'a pas été initialisé. GameLift
- GAME_SESSION_ENDED_FAILED — Le SDK Amazon GameLift Server n'a pas pu contacter le service pour signaler la fin de la session de jeu.
- GAME_SESSION_NOT_READY — La session de jeu Amazon GameLift Server n'a pas été activée.
- GAME_SESSION_READY_FAILED — Le SDK Amazon GameLift Server n'a pas pu contacter le service pour signaler que la session de jeu est prête.
- INITIALIZATION_MISMATCH — Une méthode cliente a été appelée après Server : :Initialize (), ou vice versa.
- NOT_INITIALIZED — Le GameLift serveur ou le client Amazon n'a pas été initialisé avec Initialize ().
- NO_TARGET_ALIASID_SET — Aucun alias cible n'a été défini.
- NO_TARGET_FLEET_SET — Aucun parc cible n'a été défini.
- PROCESS_ENDING_FAILED — Le SDK Amazon GameLift Server n'a pas pu contacter le service pour signaler la fin du processus.
- PROCESS_NOT_ACTIVE — Le processus serveur n'est pas encore actif, n'est pas lié à un et ne peut pas GameSession accepter ou traiter. PlayerSessions
- PROCESS_NOT_READY — Le processus du serveur n'est pas encore prêt à être activé.
- PROCESS_READY_FAILED — Le SDK Amazon GameLift Server n'a pas pu contacter le service pour signaler que le processus est prêt.
- SDK_VERSION_DETECTION_FAILED — La détection de la version du SDK a échoué.
- STX_CALL_FAILED — Un appel au composant principal du serveur xSTX a échoué.
- STX_INITIALIZATION_FAILED — Le composant principal du serveur xSTX n'a pas pu être initialisé.
- UNEXPECTED_PLAYER_SESSION — Le serveur a rencontré une session de joueur non enregistrée.

- ÉCHEC DE WEBSOCKET_CONNECT_FR
- WEBSOCKET_CONNECT_FAILURE_FORBIDDEN
- WEBSOCKET_CONNECT_FAILURE_URL INVALIDE
- DÉLAI D'ÉCHEC DE WEBSOCKET_CONNECT_FAILURE_OUT
- WEBSOCKET_RETRIABLE_SEND_MESSAGE_FAILURE — Échec récupérable lors de l'envoi d'un message au service. GameLift WebSocket
- WEBSOCKET_SEND_MESSAGE_FAILURE — Échec de l'envoi d'un message au service. GameLift WebSocket
- MATCH_BACKFILL_REQUEST_VALIDATION — La validation de la demande a échoué.
- PLAYER_SESSION_REQUEST_VALIDATION — La validation de la demande a échoué.

PlayerSessionCreationPolicy

Valeur de chaîne indiquant si la session de jeu accepte ou non de nouveaux joueurs. Les valeurs valides sont les suivantes :

- ACCEPT_ALL — Accepte toutes les sessions des nouveaux joueurs.
- DENY_ALL — Refuser toutes les sessions des nouveaux joueurs.
- NOT_SET — La session de jeu n'est pas configurée pour accepter ou refuser les sessions de nouveaux joueurs.

PlayerSessionStatus

- ACTIF
- TERMINÉ
- NOT_SET
- RÉSERVÉ
- EXPIRÉE

Référence GameLift du SDK Amazon Server 4.x pour C#

Cette référence au SDK Amazon GameLift C# Server 4.x peut vous aider à préparer votre jeu multijoueur en vue de son utilisation avec Amazon. GameLift Pour plus d'informations sur le processus d'intégration, consultez Ajoutez Amazon GameLift à votre serveur de jeu.

Rubriques

- Référence GameLift du SDK Amazon Server (C#) : Actions

- Référence GameLift du SDK Amazon Server (C#) : types de données

Référence GameLift du SDK Amazon Server (C#) : Actions

Vous pouvez utiliser cette référence du SDK pour serveur Amazon GameLift C# pour vous aider à préparer votre jeu multijoueur en vue de son utilisation avec Amazon. GameLift Pour plus de détails sur le processus d'intégration, consultezAjoutez Amazon GameLift à votre serveur de jeu.

- Actions
- Types de données

AcceptPlayerSession()

Informe le GameLift service Amazon qu'un joueur avec l'identifiant de session de joueur spécifié s'est connecté au processus du serveur et doit être validé. Amazon GameLift vérifie que l'identifiant de session du joueur est valide, c'est-à-dire que l'identifiant du joueur a réservé un emplacement de joueur dans la session de jeu. Une fois la validation effectuée, GameLift Amazon fait passer le statut de la machine à sous RÉSERVÉ à ACTIVE.

Syntaxe

```
GenericOutcome AcceptPlayerSession(String playerSessionId)
```

Paramètres

playerSessionId

Identifiant unique émis par Amazon GameLift lors de la création d'une nouvelle session de joueur. Un identifiant de session de joueur est spécifié dans un `PlayerSession` objet, qui est renvoyé en réponse à un appel du client aux actions de l'GameLiftAPI StartGameSessionPlacementCreateGameSession, DescribeGameSessionPlacement, ou DescribePlayerSessions.

Type : String

Obligatoire : oui

Valeur renvoyée

Renvoie un résultat générique composé d'un succès ou d'un échec avec un message d'erreur.

Exemple

Cet exemple illustre une fonction pour le traitement d'une demande de connexion, y compris la validation et le rejet des ID de session joueur non valides.

```
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()

Indique au GameLift service Amazon que le processus du serveur a activé une session de jeu et qu'il est désormais prêt à recevoir les connexions des joueurs. Cette action doit être appelée dans le cadre de la fonction de rappel `onStartGameSession()`, une fois que l'initialisation de toutes les sessions de jeu est terminée.

Syntaxe

```
GenericOutcome ActivateGameSession()
```

Paramètres

Cette action n'a aucun paramètre.

Valeur renvoyée

Renvoie un résultat générique composé d'un succès ou d'un échec avec un message d'erreur.

Exemple

Cet exemple illustre l'appel de `ActivateGameSession()` dans le cadre de la fonction déléguée `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()

Récupère les données de session de joueur, y compris les paramètres, les métadonnées de session et les données de joueur. Utilisez cette action pour obtenir des informations pour une seule session de joueur, pour toutes les sessions de joueur d'une session de jeu ou pour toutes les sessions de joueur associées à un seul ID de joueur.

Syntaxe

```
DescribePlayerSessionsOutcome DescribePlayerSessions(DescribePlayerSessionsRequest
 describePlayerSessionsRequest)
```

Paramètres

describePlayerSessionsDemande

Objet DescribePlayerSessionsRequest décrivant les sessions de joueur à récupérer.

Obligatoire : oui

Valeur renvoyée

En cas de réussite, renvoie un objet `DescribePlayerSessionsOutcome` qui contient un ensemble d'objets de session de joueur correspondant aux paramètres de la demande. Les objets de session du joueur ont une structure identique au type de PlayerSessiondonnées de l'GameLiftAPI Amazon du AWS SDK.

Exemple

Cet exemple illustre une demande de toutes les sessions de joueur activement connectées à une session de jeu spécifiée. En omettant NextTokenet en définissant la valeur limite sur 10, Amazon GameLift renverra les 10 premiers enregistrements de sessions des joueurs correspondant à la demande.

```
// 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()

Extrait l'ID de la session de jeu actuellement hébergée par le processus serveur, si ce dernier est actif.

Pour les processus inactifs qui ne sont pas encore activés lors d'une session de jeu, l'appel renvoie `Success` `GameSessionId` = `True` et = `""` (une chaîne vide).

Syntaxe

```
AwsStringOutcome GetGameSessionId()
```

Paramètres

Cette action n'a aucun paramètre.

Valeur renvoyée

En cas de réussite, renvoie l'ID de session de jeu en tant qu'objet `AwsStringOutcome`. En cas d'échec, renvoie un message d'erreur.

Exemple

```
var getGameSessionIdOutcome = GameLiftServerAPI.GetGameSessionId();
```

GetInstanceCertificate()

Récupère l'emplacement du fichier d'un certificat TLS codé PEM associé au parc et à ses instances. AWS Certificate Managergénère ce certificat lorsque vous créez une nouvelle flotte avec la configuration du certificat définie sur GENERATED. Utilisez ce certificat pour établir une connexion sécurisée avec un client de jeu et pour chiffrer la communication client/serveur.

Syntaxe

```
GetInstanceCertificateOutcome GetInstanceCertificate();
```

Paramètres

Cette action n'a aucun paramètre.

Valeur renvoyée

En cas de succès, renvoie un `GetInstanceCertificateOutcome` objet contenant l'emplacement du fichier de certificat TLS et de la chaîne de certificats du parc, qui sont stockés sur l'instance. Un fichier de certificat racine, extrait de la chaîne de certificats, est également stocké sur l'instance. En cas d'échec, renvoie un message d'erreur.

Pour plus d'informations sur le certificat et les données de la chaîne de certificats, consultez la section Éléments de GetCertificate réponse dans la référence de l'AWS Certificate ManagerAPI.

Exemple

```
var getInstanceCertificateOutcome = GameLiftServerAPI.GetInstanceCertificate();
```

GetSdkVersion()

Renvoie le numéro de version actuel du kit SDK intégré dans le processus serveur.

Syntaxe

```
AwsStringOutcome GetSdkVersion()
```

Paramètres

Cette action n'a aucun paramètre.

Valeur renvoyée

En cas de réussite, renvoie la version actuelle du kit SDK en tant qu'objet `AwsStringOutcome`. La chaîne renvoyée inclut uniquement le numéro de version (par exemple « 3.1.5 »). En cas d'échec, renvoie un message d'erreur.

Exemple

```
var getSdkVersionOutcome = GameLiftServerAPI.GetSdkVersion();
```

GetTerminationTime()

Renvoie l'heure d'arrêt planifiée pour un processus serveur, si une heure de résiliation est disponible. Un processus serveur effectue cette action après avoir reçu un `onProcessTerminate()` rappel du GameLift service Amazon. Amazon GameLift peut appeler `onProcessTerminate()` pour les raisons suivantes : (1) en cas de problème de santé (le processus du serveur a signalé l'intégrité du port ou n'a pas répondu à Amazon)GameLift, (2) lors de la mise hors service de l'instance lors d'un événement de réduction de la taille ou (3) lorsqu'une instance est fermée en raison d'une interruption d'une instance ponctuelle.

Si le processus a reçu un `onProcessTerminate()` rappel, la valeur renvoyée est l'heure de fin estimée. Si le processus n'a pas reçu de `onProcessTerminate()` rappel, un message d'erreur est renvoyé. En savoir plus sur l'arrêt d'un processus serveur.

Syntaxe

```
AwsDateTimeOutcome GetTerminationTime()
```

Paramètres

Cette action n'a aucun paramètre.

Valeur renvoyée

En cas de succès, renvoie l'heure de fin sous forme d'`AwsDateTimeOutcome`objet. La valeur est l'heure de fin, exprimée en ticks écoulés depuis 0001 00:00:00. Par exemple, la valeur date/heure

2020-09-13 12:26:40 -000Z est égale à 637355968000000000 ticks. Si aucune heure de fin n'est disponible, renvoie un message d'erreur.

Exemple

```
var getTerminationTimeOutcome = GameLiftServerAPI.GetTerminationTime();
```

InitSDK()

Initialise le GameLift SDK Amazon. Cette méthode doit être appelée au lancement, avant toute autre initialisation GameLift liée à Amazon.

Syntaxe

```
InitSDKOutcome InitSDK()
```

Paramètres

Cette action n'a aucun paramètre.

Valeur renvoyée

En cas de succès, renvoie un InitSdkOutcome objet indiquant que le processus serveur est prêt à être appeléProcessReady().

Exemple

```
var initSDKOutcome = GameLiftServerAPI.InitSDK();
```

ProcessEnding()

Informe le GameLift service Amazon que le processus du serveur est en train de s'arrêter. Cette méthode doit être appelée après toutes les autres tâches de nettoyage, y compris l'arrêt de toutes les sessions de jeu actives. Cette méthode doit quitter avec un code de sortie 0 ; un code de sortie différent de 0 génère un message d'événement indiquant que le processus ne s'est pas fermé correctement.

Une fois que la méthode se termine avec un code de 0, vous pouvez terminer le processus avec un code de sortie réussi. Vous pouvez également quitter le processus avec un code d'erreur. Si vous

quittez le site avec un code d'erreur, l'événement du parc indiquera que le processus s'est terminé de manière anormale (`SERVER_PROCESS_TERMINATED_UNHEALTHY`).

Syntaxe

```
GenericOutcome ProcessEnding()
```

Paramètres

Cette action n'a aucun paramètre.

Valeur renvoyée

Renvoie un résultat générique composé d'un succès ou d'un échec avec un message d'erreur.

Exemple

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

ProcessReady()

Indique au GameLift service Amazon que le processus du serveur est prêt à héberger des sessions de jeu. Appelez cette méthode après avoir invoqué InitSDK() et terminé avec succès les tâches de configuration requises pour que le processus serveur puisse héberger une session de jeu. Cette méthode ne doit être appelée qu'une seule fois par processus.

Syntaxe

```
GenericOutcome ProcessReady(ProcessParameters processParameters)
```

Paramètres

processParameters

Objet ProcessParameters communiquant les informations suivantes relatives au processus serveur :

- Noms des méthodes de rappel, implémentées dans le code du serveur de jeu, que le GameLift service Amazon invoque pour communiquer avec le processus du serveur.
- Numéro de port sur lequel le processus serveur écoute.
- Chemin d'accès à tous les fichiers spécifiques à une session de jeu que vous souhaitez qu'Amazon capture GameLift et stocke.

Obligatoire : oui

Valeur renvoyée

Renvoie un résultat générique composé d'un succès ou d'un échec avec un message d'erreur.

Exemple

Cet exemple illustre les implémentations de l'appel ProcessReady() et de la fonction déléguée.

```
// 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()

Informe le GameLift service Amazon qu'un joueur avec l'identifiant de session de joueur spécifié s'est déconnecté du processus serveur. En réponse, Amazon GameLift change le créneau de joueur pour qu'il soit disponible, ce qui permet de l'attribuer à un nouveau joueur.

Syntaxe

```
GenericOutcome RemovePlayerSession(String playerSessionId)
```

Paramètres

playerSessionId

Identifiant unique émis par Amazon GameLift lors de la création d'une nouvelle session de joueur. Un identifiant de session de joueur est spécifié dans un `PlayerSession` objet, qui est renvoyé en réponse à un appel du client aux actions de l'GameLiftAPI StartGameSessionPlacementCreateGameSession, DescribeGameSessionPlacement, ou DescribePlayerSessions.

Type : String

Obligatoire : oui

Valeur renvoyée

Renvoie un résultat générique composé d'un succès ou d'un échec avec un message d'erreur.

Exemple

```
Aws::GameLift::GenericOutcome disconnectOutcome =
    Aws::GameLift::Server::RemovePlayerSession(playerSessionId);
```

StartMatchBackfill()

Envoie une demande de recherche de nouveaux joueurs pour des emplacements ouverts dans une session de jeu créée avec FlexMatch. Voir également l'action du AWS SDK StartMatchBackfill(). Avec cette action, les requêtes de renvoi de correspondance peuvent être initiées par processus de serveur de jeu qui héberge la session de jeu. En savoir plus sur la fonction de FlexMatch remblayage.

Cette action est asynchrone. Si de nouveaux joueurs sont jumelés avec succès, le GameLift service Amazon fournit des données de matchmaking mises à jour à l'aide de la fonction de rappel. `OnUpdateGameSession()`

Un processus de serveur ne peut comporter qu'une seule requête de renvoi de correspondance à la fois. Pour envoyer une nouvelle requête, appelez d'abord StopMatchBackfill() pour annuler la requête d'origine.

Syntaxe

```
StartMatchBackfillOutcome StartMatchBackfill (StartMatchBackfillRequest
 startBackfillRequest);
```

Paramètres

StartMatchBackfillRequest

Objet StartMatchBackfillRequest qui communique les informations suivantes :

- ID de ticket à attribuer à la requête de renvoi. Ces informations sont facultatives ; si aucun identifiant n'est fourni, Amazon en GameLift générera un automatiquement.
- Matchmaker auquel envoyer la requête. L'ARN de configuration complet est obligatoire. Cette valeur peut être acquise à partir des données matchmaker de la session de jeu.
- ID de la session de jeu en cours de renvoi.
- Données de correspondance disponibles pour les joueurs actuels de la session de jeu.

Obligatoire : oui

Valeur renvoyée

Renvoie un StartMatchBackfillOutcome objet avec l'identifiant du ticket de remplissage du match ou un échec avec un message d'erreur.

Exemple

```
// 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()

Annule une requête de renvoi de correspondance active qui a été créée avec StartMatchBackfill(). Voir également l'action du AWS SDK StopMatchmaking(). En savoir plus sur la fonction de FlexMatch remblayage.

Syntaxe

```
GenericOutcome StopMatchBackfill (StopMatchBackfillRequest stopBackfillRequest);
```

Paramètres

StopMatchBackfillRequest

Objet StopMatchBackfillRequest qui identifie le ticket de correspondance à annuler :

- Identifiant de ticket attribué à la requête de renvoi en cours d'annulation
- matchmaker auquel a été envoyée la requête
- session de jeu associée à la requête de renvoi

Obligatoire : oui

Valeur renvoyée

Renvoie un résultat générique composé d'un succès ou d'un échec avec un message d'erreur.

Exemple

```
// 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()

Cette méthode est obsolète depuis la version 4.0.1. Au lieu de cela, le processus du serveur doit appeler ProcessEnding() après la fin d'une session de jeu.

Informe le GameLift service Amazon que le processus du serveur a mis fin à la session de jeu en cours. Cette action est appelée lorsque le processus du serveur reste actif et prêt à héberger une nouvelle session de jeu. Il ne doit être appelé qu'une fois la procédure de fin de session de jeu terminée, car il indique à Amazon GameLift que le processus du serveur est immédiatement disponible pour héberger une nouvelle session de jeu.

Cette action n'est pas appelée si le processus du serveur doit être arrêté après la fin de la session de jeu. Appelez plutôt ProcessEnding() pour signaler que la session de jeu et le processus du serveur se terminent.

Syntaxe

```
GenericOutcome TerminateGameSession()
```

Paramètres

Cette action n'a aucun paramètre.

Valeur renvoyée

Renvoie un résultat générique composé d'un succès ou d'un échec avec un message d'erreur.

Exemple

Cet exemple illustre un processus de serveur à la fin d'une session de jeu.

```
// 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()

Met à jour la capacité de la session de jeu à accepter de nouvelles sessions de joueur. Une session de jeu peut être définie pour accepter ou refuser toutes les nouvelles sessions joueur. (Voir également l'action UpdateGameSession() dans la référence de l'API Amazon GameLift Service).

Syntaxe

```
GenericOutcome UpdatePlayerSessionCreationPolicy(PlayerSessionCreationPolicy
 playerSessionPolicy)
```

Paramètres

newPlayerSessionPolitique

Valeur de chaîne indiquant si la session de jeu accepte ou non de nouveaux joueurs.

Type : enum PlayerSessionCreationPolicy. Les valeurs valides sont les suivantes :

- ACCEPT_ALL — Accepte toutes les sessions des nouveaux joueurs.
- DENY_ALL — Refuser toutes les sessions des nouveaux joueurs.

Obligatoire : oui

Valeur renvoyée

Renvoie un résultat générique composé d'un succès ou d'un échec avec un message d'erreur.

Exemple

Cet exemple définit la stratégie de participation de la session de jeu actuelle de manière à ce que tous les joueurs soient acceptés.

```
var updatePlayerSessionCreationPolicyOutcomex =

 GameLiftServerAPI.UpdatePlayerSessionCreationPolicy(PlayerSessionCreationPolicy.ACCEPT_ALL);
```

Référence GameLift du SDK Amazon Server (C#) : types de données

Vous pouvez utiliser cette référence du SDK pour serveur Amazon GameLift C# pour vous aider à préparer votre jeu multijoueur en vue de son utilisation avec Amazon. GameLift Pour plus de détails sur le processus d'intégration, consultezAjoutez Amazon GameLift à votre serveur de jeu.

- Actions
- Types de données

LogParameters

Ce type de données est utilisé pour identifier les fichiers générés pendant une session de jeu que vous souhaitez qu'Amazon GameLift télécharge et stocke une fois la session de jeu terminée. Ces informations sont communiquées au GameLift service Amazon lors d'un ProcessReady() appel.

Table des matières

logPaths

Liste des chemins de répertoire vers les fichiers journaux du serveur de jeu que vous souhaitez GameLift qu'Amazon stocke pour un accès ultérieur. Ces fichiers sont générés par un processus

serveur pendant chaque session de jeu ; les noms et les chemins des fichiers sont définis dans votre serveur de jeux et stockés dans le répertoire racine de la version de génération de jeu. Les chemins du journal doivent être absolus. Par exemple, si la version de génération de votre jeu stocke les journaux de session de jeu suivant un chemin tel que `MyGame\sessionlogs\`, le chemin d'accès aux journaux est `c:\game\MyGame\sessionLogs` (sur une instance Windows) ou `/local/game/MyGame/sessionLogs` (sur une instance Linux).

Type : Liste<String>

Obligatoire : non

DescribePlayerSessionsRequest

Ce type de données est utilisé pour spécifier les sessions de joueur à récupérer. Il peut être utilisé de plusieurs manières : (1) PlayerSessionId pour demander une session de joueur spécifique ; (2) GameSessionId pour demander toutes les sessions des joueurs pendant la session de jeu spécifiée ; ou (3) PlayerId pour demander toutes les sessions des joueurs pour le joueur spécifié. Pour les volumes importants de sessions de joueur, utilisez les paramètres de pagination pour récupérer les résultats en tant que pages séquentielles.

Table des matières

GameSessionId

Identifiant de session de jeu unique. Utilisez ce paramètre pour demander toutes les sessions de joueur pour la session de jeu spécifiée. Le format de l'ID de session de jeu est le suivant : `arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string>`. La valeur de <ID string>peut être une chaîne d'ID personnalisée (si spécifiée lors de la création de la session de jeu) ou une chaîne générée automatiquement.

Type : chaîne

Obligatoire : non

Limite

Nombre maximum de résultats à renvoyer. Utilisez ce paramètre avec NextTokenpour obtenir des résultats sous la forme d'un ensemble de pages séquentielles. Si un ID de session de joueur est spécifié, ce paramètre est ignoré.

Type : entier

Obligatoire : non

NextToken

Jeton indiquant le début de la prochaine page séquentielle de résultats. Utilisez le jeton qui est renvoyé par un appel précédent à cette action. Pour spécifier le début de l'ensemble de résultats, ne spécifiez aucune valeur. Si un ID de session de joueur est spécifié, ce paramètre est ignoré.

Type : chaîne

Obligatoire : non

PlayerId

Identifiant unique pour un joueur. Les ID de joueur sont définis par le développeur. Consultez Générer des identifiants de joueurs.

Type : chaîne

Obligatoire : non

PlayerSessionId

Identifiant unique d'une session de joueur.

Type : chaîne

Obligatoire : non

PlayerSessionStatusFilter

État de session de joueur pour filtrer les résultats. Les états de session de joueur possibles sont les suivants :

- RESERVED - La demande de session de joueur a été reçue, mais le joueur ne s'est pas encore connecté au processus serveur et/ou n'a pas encore été validé.
- ACTIVE - Le joueur a été validé par le processus serveur et est actuellement connecté.
- COMPLETED - La connexion du joueur a été abandonnée.
- TIMEDOUT - Une demande de session de joueur a été reçue, mais le joueur ne s'est pas connecté et/ou n'a pas été validé avant l'expiration du délai (60 secondes).

Type : chaîne

Obligatoire : non

ProcessParameters

Ce type de données contient l'ensemble des paramètres envoyés au GameLift service Amazon lors d'un ProcessReady() appel.

Table des matières

port

Numéro de port sur lequel le processus serveur écoute les nouvelles connexions de joueur. La valeur doit être comprise dans la plage de ports configurée pour toutes les flottes déployant cette version de génération du serveur de jeux. Ce numéro de port est inclus dans les objets de session de jeu et de session de joueur, que les sessions de jeu utilisent pour se connecter à un processus serveur.

Type : entier

Obligatoire : oui

logParameters

Objet comportant une liste de chemins de répertoires vers les fichiers journaux de sessions de jeu.

Type: `Aws::GameLift::Server::LogParameters`

Obligatoire : oui

onStartGameSéance

Nom de la fonction de rappel invoquée par le GameLift service Amazon pour activer une nouvelle session de jeu. Amazon GameLift appelle cette fonction en réponse à la demande du client CreateGameSession. La fonction de rappel prend un GameSessionobjet (défini dans la référence de l'API Amazon GameLift Service).

Type: `void OnStartGameSessionDelegate(GameSession gameSession)`

Obligatoire : oui

onProcessTerminate

Nom de la fonction de rappel invoquée par le GameLift service Amazon pour forcer l'arrêt du processus du serveur. Après avoir appelé cette fonction, Amazon GameLift attend cinq minutes que le processus serveur s'arrête et répond par un ProcessEnding() appel avant d'arrêter le processus serveur.

Type: `void OnProcessTerminateDelegate()`

Obligatoire : oui

onHealthCheck

Nom de la fonction de rappel invoquée par le GameLift service Amazon pour demander un rapport d'état de santé au processus serveur. Amazon GameLift appelle cette fonction toutes les 60 secondes. Après avoir appelé cette fonction, Amazon GameLift attend une réponse pendant 60 secondes et, si aucune réponse n'est reçue, enregistre le processus du serveur comme étant défectueux.

Type: `bool OnHealthCheckDelegate()`

Obligatoire : oui

onUpdateGameSéance

Nom de la fonction de rappel invoquée par le GameLift service Amazon pour transmettre un objet de session de jeu mis à jour au processus du serveur. Amazon GameLift fait appel à cette fonction lorsqu'une demande de remplissage de correspondance a été traitée afin de fournir des données de matchmaking mises à jour. Il transmet un GameSessionobjet, une mise à jour de statut (`updateReason`) et l'identifiant du ticket de remplissage du match.

Type: `void OnUpdateGameSessionDelegate ( UpdateGameSession updateGameSession )`

Obligatoire : non

StartMatchBackfillRequest

Ce type de données est utilisé pour envoyer une requête de renvoi de correspondance. Les informations sont communiquées au GameLift service Amazon lors d'un StartMatchBackfill() appel.

Table des matières

GameSessionArn

Identifiant de session de jeu unique. La méthode SDK GetGameSessionId() renvoie l'identifiant au format ARN.

Type : String

Obligatoire : oui

MatchmakingConfigurationArn

Identifiant unique, sous la forme d'un ARN, pour le matchmaker à utiliser pour cette requête. Pour trouver le matchmaker qui a été utilisé pour créer la session de jeu d'origine, recherchez dans l'objet de session de jeu, dans la propriété de données de matchmaker. Pour en savoir plus sur les données de matchmaking, consultez la section Travailler avec les données de matchmaking.

Type : String

Obligatoire : oui

Joueurs

Ensemble de données représentant tous les joueurs qui sont actuellement dans la session de jeu. Le matchmaker utilise ces informations pour rechercher de nouveaux joueurs qui constituent de bonnes correspondances pour les joueurs actuels. Consultez le guide de référence des GameLift API Amazon pour obtenir une description du format d'objet Player. Pour trouver des attributs de joueur, des identifiants et des affectations d'équipe, recherchez dans l'objet de session de jeu, dans la propriété des données de matchmaker. Si une latence est utilisée par le matchmaker, collectez la latence mise à jour pour la région actuelle et incluez-la dans les données de chaque joueur.

Type : Player[ ]

Obligatoire : oui

TicketId

Identifiant unique pour une correspondance ou un ticket de requête de renvoi de correspondance. Si aucune valeur n'est fournie ici, Amazon en GameLift générera une sous la forme d'un UUID. Utilisez cet identifiant pour suivre l'état du ticket de renvoi de correspondance ou annuler la requête si nécessaire.

Type : chaîne

Obligatoire : non

StopMatchBackfillRequest

Ce type de données est utilisé pour annuler une demande de renvoi de correspondance. Les informations sont communiquées au GameLift service Amazon lors d'un StopMatchBackfill() appel.

Table des matières

GameSessionArn

Identifiant de session de jeu unique associé à la requête en cours d'annulation.

Type : String

Obligatoire : oui

MatchmakingConfigurationArn

Identifiant unique du matchmaker auquel cette requête a été envoyée.

Type : String

Obligatoire : oui

TicketId

Identifiant unique d'un ticket de requête de correspondance à annuler.

Type : String

Obligatoire : oui

Référence GameLift du SDK Amazon Server pour Go

Vous pouvez utiliser cette référence du SDK du serveur Amazon GameLift Go pour vous aider à préparer votre jeu multijoueur en vue de son utilisation avec AmazonGameLift. Pour plus de détails sur le processus d'intégration, consultezAjoutez Amazon GameLift à votre serveur de jeu.

Rubriques

- Référence GameLift du SDK (Go) du serveur Amazon : Actions
- Référence GameLift du SDK Amazon Server (Go) : types de données

Référence GameLift du SDK (Go) du serveur Amazon : Actions

Vous pouvez utiliser cette référence du SDK du serveur Amazon GameLift Go pour vous aider à préparer votre jeu multijoueur en vue de son utilisation avec Amazon GameLift. Pour plus de détails sur le processus d'intégration, consultezAjoutez Amazon GameLift à votre serveur de jeu.

`GameLiftServerAPI.go`définit les actions du SDK du serveur Go.

Actions

- GetSdkVersion()
- InitSDK()
- ProcessReady()
- ProcessEnding()
- ActivateGameSession()
- UpdatePlayerSessionCreationPolicy()
- GetGameSessionId()
- GetTerminationTime()
- AcceptPlayerSession()
- RemovePlayerSession()
- DescribePlayerSessions()
- StartMatchBackfill()
- StopMatchBackfill()
- GetComputeCertificate()
- GetFleetRoleCredentials()
- Détruire ()

GetSdkVersion()

Renvoie le numéro de version actuel du kit SDK intégré dans le processus serveur.

Syntaxe

```
func GetSdkVersion() (string, error)
```

Valeur renvoyée

En cas de succès, renvoie la version actuelle du SDK sous forme de chaîne. La chaîne renvoyée inclut le numéro de version (exemple`5.0.0`). En cas d'échec, renvoie un message d'erreur tel que`common.SdkVersionDetectionFailed`.

Exemple

```
version, err := server.GetSdkVersion()
```

InitSDK()

Initialise le GameLift SDK Amazon. Appelez cette méthode au lancement avant toute autre initialisation liée à Amazon GameLift . Cette méthode permet de configurer la communication entre le serveur et le GameLift service Amazon.

Syntaxe

```
func InitSDK(params ServerParameters) error
```

Paramètres

ServerParameters

Pour initialiser un serveur de jeu sur une GameLift Anywhere flotte Amazon, créez un `ServerParameters` objet avec les informations suivantes :

- URL WebSocket utilisée pour vous connecter à votre serveur de jeu.
- ID du processus utilisé pour héberger votre serveur de jeu.
- L'ID de l'ordinateur hébergeant les processus de votre serveur de jeu.
- L'ID de la GameLift flotte Amazon contenant votre GameLift Anywhere ordinateur Amazon.
- Le jeton d'autorisation généré par l' GameLift opération Amazon.

Pour initialiser un serveur de jeu sur une flotte EC2 GameLift gérée par Amazon, créez un `ServerParameters` objet sans paramètres. Avec cet appel, l' GameLift agent Amazon configure l'environnement informatique et se connecte automatiquement au GameLift service Amazon pour vous.

Valeur renvoyée

En cas de succès, renvoie `nil` une erreur indiquant que le processus serveur est prêt à être appeléProcessReady().

> **Note**
>
> Si les appels à échouent pour `InitSDK()` les builds de jeu déployés sur des flottes Anywhere, vérifiez le `ServerSdkVersion` paramètre utilisé lors de la création de la ressource de build. Vous devez définir explicitement cette valeur en fonction de la version du SDK du serveur utilisée. La valeur par défaut de ce paramètre est 4.x, ce qui n'est pas

compatible. Pour résoudre ce problème, créez une nouvelle version et déployez-la sur une nouvelle flotte.

Exemple

GameLift AnywhereExemple 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)
```

Exemple d'EC2 GameLift géré par 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()

Indique à Amazon GameLift que le processus du serveur est prêt à héberger des sessions de jeu. Appelez cette méthode après l'avoir invoquée. InitSDK() Cette méthode ne doit être appelée qu'une seule fois par processus.

Syntaxe

```
func ProcessReady(param ProcessParameters) error
```

Paramètres

ProcessParameters

Un ProcessParameters objet communique les informations suivantes concernant le processus du serveur :

- Les noms des méthodes de rappel implémentées dans le code du serveur de jeu invoqué par le GameLift service Amazon pour communiquer avec le processus du serveur.
- Numéro de port sur lequel le processus serveur est en train d'écouter.
- Type de LogParameters données contenant le chemin d'accès à tous les fichiers spécifiques à une session de jeu que vous souhaitez qu'Amazon capture et GameLift stocke.

Valeur renvoyée

Renvoie une erreur accompagnée d'un message d'erreur en cas d'échec de la méthode. Renvoie `nil` si la méthode est réussie.

Exemple

Cet exemple illustre les implémentations de l'appel ProcessReady() et de la fonction déléguée.

```
// 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()

Indique à Amazon GameLift que le processus du serveur est en train de se terminer.
Appelez cette méthode après toutes les autres tâches de nettoyage (y compris la fermeture

de la session de jeu active) et avant de terminer le processus. En fonction du résultat de`ProcessEnding()`, le processus se termine avec succès (0) ou erreur (-1) et génère un événement de flotte. Si le processus se termine par une erreur, l'événement de flotte généré est`SERVER_PROCESS_TERMINATED_UNHEALTHY`.

Syntaxe

```
func ProcessEnding() error
```

Valeur renvoyée

Renvoie un code d'erreur 0 ou un code d'erreur défini.

Exemple

```
// 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()

Informe Amazon GameLift que le processus du serveur a activé une session de jeu et qu'il est désormais prêt à recevoir les connexions des joueurs. Cette action est appelée dans le cadre de la fonction de rappel `onStartGameSession()`, après toute initialisation de session de jeu.

Syntaxe

```
func ActivateGameSession() error
```

Valeur renvoyée

Renvoie une erreur accompagnée d'un message d'erreur en cas d'échec de la méthode.

Exemple

Cet exemple montre l'`ActivateGameSession()`appel dans le cadre de la fonction de `onStartGameSession()` délégation.

```
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()

Met à jour la capacité de la session de jeu à accepter de nouvelles sessions de joueur. Une session de jeu peut être définie pour accepter ou refuser toutes les nouvelles sessions joueur.

Syntaxe

```
func UpdatePlayerSessionCreationPolicy(policy model.PlayerSessionCreationPolicy) error
```

Paramètres

playerSessionCreationPolitique

Valeur de chaîne indiquant si la session de jeu accepte de nouveaux joueurs.

Les valeurs valides sont les suivantes :

- **`model.AcceptAll`**— Acceptez toutes les sessions pour nouveaux joueurs.
- **`model.DenyAll`**— Refusez toutes les sessions pour les nouveaux joueurs.

Valeur renvoyée

Renvoie une erreur avec un message d'erreur en cas d'échec.

Exemple

Cet exemple définit la stratégie de participation de la session de jeu actuelle de manière à ce que tous les joueurs soient acceptés.

```
err := server.UpdatePlayerSessionCreationPolicy(model.AcceptAll)
```

GetGameSessionId()

Récupère l'ID de la session de jeu hébergée par le processus serveur actif.

Syntaxe

```
func GetGameSessionID() (string, error)
```

Paramètres

Cette action n'a aucun paramètre.

Valeur renvoyée

En cas de succès, renvoie l'identifiant de session de jeu et aucune erreur. Pour les processus inactifs qui ne sont pas encore activés lors d'une session de jeu, l'appel renvoie une chaîne vide et une `nil` erreur.

Exemple

```
gameSessionID, err := server.GetGameSessionID()
```

GetTerminationTime()

Renvoie l'heure à laquelle il est prévu d'arrêter un processus serveur si une heure de fin est disponible. Un processus serveur exécute cette action après avoir reçu un `onProcessTerminate()` rappel d'Amazon GameLift. Amazon GameLift appelle `onProcessTerminate()` pour les raisons suivantes :

- Lorsque le processus du serveur a signalé un problème de santé ou n'a pas répondu à Amazon GameLift.
- Lorsque vous mettez fin à l'instance lors d'un événement de réduction de la taille.
- Lorsqu'une instance est interrompue en raison d'une interruption ponctuelle.

Syntaxe

```
func GetTerminationTime() (int64, error)
```

Valeur renvoyée

En cas de succès, renvoie l'horodatage en secondes pendant lequel le processus du serveur est programmé pour s'arrêter et une `nil` erreur se termine. La valeur est le délai de résiliation, exprimé en ticks écoulés depuis. `0001 00:00:00` Par exemple, la valeur de la date et de l'heure `2020-09-13 12:26:40 -000Z` est égale à celle `637355968000000000` des ticks. Si aucune heure de résiliation n'est disponible, renvoie un message d'erreur.

Exemple

```
terminationTime, err := server.GetTerminationTime()
```

AcceptPlayerSession()

Informe Amazon GameLift qu'un joueur possédant l'identifiant de session de joueur spécifié s'est connecté au processus du serveur et doit être validé. Amazon GameLift vérifie que l'identifiant de session du joueur est valide. Une fois la session du joueur validée, Amazon GameLift change le statut de l'emplacement du joueur de `RESERVED` à`ACTIVE`.

Syntaxe

```
func AcceptPlayerSession(playerSessionID string) error
```

Paramètres

`playerSessionId`

ID unique émis par Amazon GameLift lors de la création d'une nouvelle session de joueur.

Valeur renvoyée

Renvoie un résultat générique consistant en un succès ou un échec avec un message d'erreur.

Exemple

Cet exemple traite une demande de connexion qui inclut la validation et le rejet d'identifiants de session de joueur non valides.

```
func ReceiveConnectingPlayerSessionID(conn Connection, playerSessionID string) {
```

```
    err := server.AcceptPlayerSession(playerSessionID)
    if err != nil {
        connection.Accept()
    } else {
        connection.Reject(err.Error())
    }
}
```

RemovePlayerSession()

Indique à Amazon GameLift qu'un joueur s'est déconnecté du processus du serveur. En réponse, Amazon GameLift modifie l'emplacement du joueur pour le rendre disponible.

Syntaxe

```
func RemovePlayerSession(playerSessionID string) error
```

Paramètres

`playerSessionId`

ID unique émis par Amazon GameLift lors de la création d'une nouvelle session de joueur.

Valeur renvoyée

Renvoie un résultat générique consistant en un succès ou un échec avec un message d'erreur.

Exemple

```
err := server.RemovePlayerSession(playerSessionID)
```

DescribePlayerSessions()

Récupère les données de session du joueur, notamment les paramètres, les métadonnées de session et les données du joueur. Utilisez cette méthode pour obtenir des informations sur les points suivants :

- Une session solo
- Toutes les sessions des joueurs au cours d'une session de jeu
- Toutes les sessions de joueur associées à un identifiant de joueur unique

Syntaxe

```
func DescribePlayerSessions(req request.DescribePlayerSessionsRequest)
 (result.DescribePlayerSessionsResult, error) {
 return srv.describePlayerSessions(&req)
}
```

Paramètres

DescribePlayerSessionsRequest

Un `DescribePlayerSessionsRequest` objet décrit les sessions de joueur à récupérer.

Valeur renvoyée

En cas de succès, renvoie un `DescribePlayerSessionsResult` objet contenant un ensemble d'objets de session de joueur correspondant aux paramètres de la demande.

Exemple

Cet exemple demande toutes les sessions de joueur activement connectées à une session de jeu spécifiée. En omettant NextTokenet en définissant la valeur limite sur 10, Amazon GameLift renvoie les 10 premiers enregistrements de session de joueur correspondant à la demande.

```
// 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()

Envoie une demande de recherche de nouveaux joueurs pour des emplacements ouverts dans une session de jeu créée avec FlexMatch. Pour plus d'informations, voir la fonction de FlexMatch remblayage.

Cette action est asynchrone. Si de nouveaux joueurs sont jumelés, Amazon GameLift fournit des données de matchmaking mises à jour à l'aide de la fonction de rappel. `OnUpdateGameSession()`

Un processus de serveur ne peut comporter qu'une seule requête de renvoi de correspondance à la fois. Pour envoyer une nouvelle requête, appelez d'abord StopMatchBackfill() pour annuler la requête d'origine.

Syntaxe

```
func StartMatchBackfill(req request.StartMatchBackfillRequest)
 (result.StartMatchBackfillResult, error)
```

Paramètres

StartMatchBackfillRequest

Un StartMatchBackfillRequest objet communique les informations suivantes :

- ID de ticket à attribuer à la requête de renvoi. Ces informations sont facultatives ; si aucun identifiant n'est fourni, Amazon en GameLift génère un.
- Matchmaker auquel envoyer la requête. L'ARN de configuration complet est obligatoire. Cette valeur se trouve dans les données du matchmaker de la session de jeu.
- ID de la session de jeu à compléter.
- Les données de matchmaking disponibles pour les joueurs actuels de la session de jeu.

Valeur renvoyée

Renvoie un `StartMatchBackfillResult` objet avec l'identifiant du ticket de remplacement correspondant, ou un échec avec un message d'erreur.

Exemple

```
// 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()

Annule une demande de remplacement de match active. Pour plus d'informations, voir la fonction de FlexMatch remblayage.

Syntaxe

```
func StopMatchBackfill(req request.StopMatchBackfillRequest) error
```

Paramètres

StopMatchBackfillRequest

Un StopMatchBackfillRequest objet qui identifie le ticket de matchmaking à annuler :

- L'identifiant du ticket attribué à la demande de remblayage.
- L'entremetteur à qui la demande de remblayage a été envoyée.
- La session de jeu associée à la demande de remplacement.

Valeur renvoyée

Renvoie un résultat générique consistant en un succès ou un échec avec un message d'erreur.

Exemple

```
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()

Récupère le chemin d'accès au certificat TLS utilisé pour chiffrer la connexion réseau entre le serveur de jeu et votre client de jeu. Vous pouvez utiliser le chemin du certificat lorsque vous enregistrez votre appareil informatique dans une GameLift Anywhere flotte Amazon. Pour plus d'informations, consultez RegisterCompute.

Syntaxe

```
func GetComputeCertificate() (result.GetComputeCertificateResult, error)
```

Valeur renvoyée

Renvoie un `GetComputeCertificateResult` objet contenant les éléments suivants :

- CertificatePath: chemin d'accès au certificat TLS sur votre ressource de calcul. Lorsque vous utilisez une flotte GameLift gérée par Amazon, ce chemin contient :
  - `certificate.pem`: le certificat de l'utilisateur final. La chaîne de certificats complète est la combinaison des `certificateChain.pem` éléments ajoutés à ce certificat.
  - `certificateChain.pem`: chaîne de certificats qui contient le certificat racine et les certificats intermédiaires.
  - `rootCertificate.pem`: le certificat racine.
  - `privateKey.pem`: clé privée pour le certificat d'utilisateur final.
- ComputeName: nom de votre ressource de calcul.

Exemple

```
tlsCertificate, err := server.GetFleetRoleCredentials(getFleetRoleCredentialsRequest)
```

GetFleetRoleCredentials()

Récupère les informations d'identification du rôle de service que vous créez pour accorder des autorisations à votre interlocuteur Services AWS sur Amazon GameLift. Ces informations

d'identification permettent à votre serveur de jeu d'utiliser vos AWS ressources. Pour plus d'informations, consultez Configurer un rôle de service IAM pour Amazon GameLift.

Syntaxe

```
func GetFleetRoleCredentials(
  req request.GetFleetRoleCredentialsRequest,
) (result.GetFleetRoleCredentialsResult, error) {
  return srv.getFleetRoleCredentials(&req)
}
```

Paramètres

GetFleetRoleCredentialsRequest

Des informations d'identification de rôle qui étendent un accès limité à vos AWS ressources au serveur de jeu.

Valeur renvoyée

Renvoie un `GetFleetRoleCredentialsResult` objet contenant les éléments suivants :

- AssumedRoleUserArn - Le nom de ressource Amazon (ARN) de l'utilisateur auquel appartient le rôle de service.
- AssumedRoleId - L'ID de l'utilisateur auquel appartient le rôle de service.
- AccessKeyId - L'ID de clé d'accès pour authentifier et fournir un accès à vos AWS ressources.
- SecretAccessKey - L'identifiant de la clé d'accès secrète pour l'authentification.
- SessionToken - Un jeton pour identifier la session active en cours qui interagit avec vos AWS ressources.
- Expiration : délai avant l'expiration des informations d'identification de votre session.

Exemple

```
// form the customer credentials request
getFleetRoleCredentialsRequest := request.NewGetFleetRoleCredentials()
getFleetRoleCredentialsRequest.RoleArn = "arn:aws:iam::123456789012:role/service-role/
exampleGameLiftAction"
```

```
credentials, err := server.GetFleetRoleCredentials(getFleetRoleCredentialsRequest)
```

Détruire ()

Libère de la mémoire le SDK du serveur de GameLift jeu Amazon. Il est recommandé d'appeler cette méthode après `ProcessEnding()` et avant de terminer le processus. Si vous utilisez une flotte Anywhere et que vous n'interrompez pas les processus du serveur après chaque session de jeu, appelez `Destroy()` puis réinitialisez avant `InitSDK()` d'informer Amazon GameLift que le processus est prêt à héberger une session de jeu avec. `ProcessReady()`

Syntaxe

```
func Destroy() error {
 return srv.destroy()
}
```

Valeur renvoyée

Renvoie une erreur accompagnée d'un message d'erreur en cas d'échec de la méthode.

Exemple

```
// 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)
  }
}
```

Référence GameLift du SDK Amazon Server (Go) : types de données

Vous pouvez utiliser cette référence du SDK du serveur Amazon GameLift Go pour vous aider à préparer votre jeu multijoueur en vue de son utilisation avec Amazon GameLift. Pour plus de détails sur le processus d'intégration, consultezAjoutez Amazon GameLift à votre serveur de jeu.

Types de données

- LogParameters
- ProcessParameters
- UpdateGameSession
- GameSession
- ServerParameters
- StartMatchBackfillRequest
- Joueur
- DescribePlayerSessionsRequest
- StopMatchBackfillRequest
- GetFleetRoleCredentialsRequest

LogParameters

Un objet identifiant les fichiers générés au cours d'une session de jeu que vous souhaitez qu'Amazon télécharge et stocke GameLift à la fin de la session de jeu. Le serveur de jeu fournit `LogParameters` à Amazon dans GameLift le cadre d'un `ProcessParameters` objet lors d'un ProcessReady() appel.

Propriétés	Description
`LogPaths`	La liste des chemins de répertoire vers les fichiers journaux du serveur de jeu que vous souhaitez qu'Amazon GameLift stocke pour un accès futur. Le processus du serveur génère ces fichiers lors de chaque session de jeu. Vous définissez les chemins et les noms des fichiers sur votre serveur de jeu et vous les stockez dans le répertoire racine du jeu. Les chemins du journal doivent être absolus. Par exemple, si votre build de jeu stocke les journaux de session de jeu dans un chemin tel que`MyGame\sessionLogs\` , alors le chemin se `c:\game\MyGame\sessionLogs` trouve sur une instance Windows. Type : `[]string` Obligatoire : non

ProcessParameters

Objet décrivant la communication entre un processus serveur et Amazon GameLift. Le processus du serveur fournit ces informations GameLift à Amazon en appelantProcessReady().

Propriétés	Description
`LogParameters`	Objet avec des chemins de répertoire vers des fichiers générés au cours d'une session de jeu. Amazon GameLift copie et stocke les fichiers pour un accès futur. Type : `LogParameters` Obligatoire : non
`OnHealthCheck`	Fonction de rappel GameLift invoquée par Amazon pour demander un rapport d'état de santé au processus du serveur. Amazon GameLift appelle cette fonction toutes les 60 secondes et attend une réponse pendant 60 secondes. Le processus du serveur revient `TRUE` s'il est sain, `FALSE` sinon sain. Si aucune réponse n'est renvoyée, Amazon considère GameLift que le processus du serveur n'est pas sain. Type : `OnHealthCheck func() bool` Obligatoire : non
`OnProcessTerminate`	La fonction de rappel GameLift invoquée par Amazon pour forcer le processus du serveur à s'arrêter. Après avoir appelé cette fonction, Amazon GameLift attend 5 minutes que le processus du serveur s'arrête et répond par un ProcessEnding() appel avant d'arrêter le processus du serveur. Type : `OnProcessTerminate func()` Obligatoire : oui
`OnStartGameSession`	Fonction de rappel GameLift invoquée par Amazon pour transmettre un objet de session de jeu mis à jour au processus du serveur. Amazon GameLift appelle cette fonction lorsqu'une demande de mise en correspondance a été traitée pour fournir des données de matchmaking mises à jour. Il transmet un

	GameSessionobjet, une mise à jour de statut (`updateReason`) et l'identifiant du ticket de remplacement des matchs. Type : `OnStartGameSession func (model.GameSession )` Obligatoire : oui
`OnUpdateGameSession`	La fonction de rappel GameLift invoquée par Amazon pour transmettre les informations de session de jeu mises à jour au processus du serveur. Amazon GameLift appelle cette fonction après avoir traité une demande de mise en correspondance afin de fournir des données de matchmaking mises à jour. Type : `OnUpdateGameSession func (model.UpdateGameSession)` Obligatoire : non
`Port`	Numéro de port sur lequel le processus serveur écoute les connexions des nouveaux joueurs. La valeur doit être comprise dans la plage de ports configurée pour toutes les flottes déployant cette version de génération du serveur de jeux. Ce numéro de port est inclus dans les objets de session de jeu et de session de joueur, que les sessions de jeu utilisent pour se connecter à un processus serveur. Type : `int` Obligatoire : oui

UpdateGameSession

Les mises à jour apportées à un objet de session de jeu, y compris la raison pour laquelle la session de jeu a été mise à jour et l'identifiant du ticket de remplissage associé si le remplissage est utilisé pour remplir les sessions des joueurs pendant la session de jeu.

Propriétés	Description
GameSession	Un GameSessionobjet défini par l' GameLift API Amazon. L'`GameSession` objet contient des propriétés décrivant une session de jeu.

Propriétés	Description
	Type : `GameSession GameSession()` Obligatoire : oui
UpdateReason	La raison pour laquelle la session de jeu est mise à jour. Type : `UpdateReason UpdateReason()` Obligatoire : oui
BackfillTicketId	L'identifiant du ticket de remplacement qui tente de mettre à jour la session de jeu. Type : `String` Obligatoire : non

GameSession

Les détails d'une session de jeu.

Propriétés	Description
GameSessionId	Identifiant unique pour la session de jeu. Une session de jeu Amazon Resource Name (ARN) a le format suivant :`arn:aws:gamelift:<region>::gamesession/<fleet ID>/<custom ID string or idempotency token>` . Type : `String` Obligatoire : non
Nom	Une étiquette descriptive de la session de jeu. Type : `String` Obligatoire : non
FleetId	Identifiant unique de la flotte sur laquelle s'exécute la session de jeu.

Propriétés	Description
	Type : `String` Obligatoire : non
MaximumPlayerSessionCount	Le nombre maximum de connexions de joueurs à la session de jeu. Type : `Integer` Obligatoire : non
Port	Le numéro de port de la session de jeu. Pour se connecter à un serveur de GameLift jeu Amazon, une application a besoin à la fois de l'adresse IP et du numéro de port. Type : `Integer` Obligatoire : non
IpAddress	Adresse IP de la session de jeu. Pour se connecter à un serveur de GameLift jeu Amazon, une application a besoin à la fois de l'adresse IP et du numéro de port. Type : `String` Obligatoire : non
GameSessionData	Ensemble de propriétés de session de jeu personnalisées, mises en forme en tant que valeur de chaîne unique. Type : `String` Obligatoire : non

Propriétés	Description
MatchmakerData	Les informations sur le processus de matchmaking qui a été utilisé pour créer la session de jeu, en syntaxe JSON, formatées sous forme de chaîne. En plus de la configuration de matchmaking utilisée, il contient des données sur tous les joueurs affectés au match, y compris les attributs des joueurs et les affectations des équipes. Type : `String` Obligatoire : non
GameProperties	Ensemble de propriétés personnalisées pour une session de jeu, formatées sous forme de paires clé:valeur. Ces propriétés sont transmises avec une demande de démarrage d'une nouvelle session de jeu. Type : `map[string] string` Obligatoire : non
DnsName	Identifiant DNS attribué à l'instance qui exécute la session de jeu. Les valeurs ont le format suivant : • Flottes compatibles TLS :. `<unique identifier>.<region identifier>.amazongamelift.com` • Flottes non compatibles TLS :. `ec2-<unique identifier>.compute.amazonaws.com` Lorsque vous vous connectez à une session de jeu exécutée sur une flotte compatible TLS, vous devez utiliser le nom DNS et non l'adresse IP. Type : `String` Obligatoire : non

ServerParameters

Informations utilisées pour maintenir la connexion entre un GameLift Anywhere serveur Amazon et le GameLift service Amazon. Ces informations sont utilisées lors du lancement de nouveaux processus

serveur avecInitSDK(). Pour les serveurs hébergés sur des instances EC2 GameLift gérées par Amazon, utilisez un objet vide.

Propriétés	Description
`WebSocket URL`	L'`GameLiftServerSdkEndpoint` Amazon GameLift revient lorsque vous `RegisterCompute` recherchez une ressource GameLift Anywhere informati que Amazon. Type : `string` Obligatoire : oui
`ProcessID`	Un identifiant unique enregistré auprès du processus serveur hébergeant votre jeu. Type : `string` Obligatoire : oui
`HostID`	Identifiant unique de la ressource de calcul hébergeant le nouveau processus du serveur. Le `HostID` est celui `ComputeName` utilisé lorsque vous avez enregistré votre ordinateur. Pour plus d'informations, consultez RegisterCompute. Type : `string` Obligatoire : oui
`FleetID`	Identifiant unique de la flotte dans laquelle le calcul est enregistré. Pour plus d'informations, consultez RegisterCompute. Type : `string` Obligatoire : oui
`AuthToken`	Le jeton d'authentification généré par Amazon GameLift qui authentifie votre serveur auprès d'Amazon GameLift. Pour plus d'informations, consultez GetComputeAuthToken.

Propriétés	Description
	Type : `string` Obligatoire : oui

StartMatchBackfillRequest

Informations utilisées pour créer une demande de remplissage par matchmaking. Le serveur de jeu communique ces informations à Amazon GameLift lors d'un StartMatchBackfill() appel.

Propriétés	Description
GameSessionArn	L'identifiant unique de session de jeu. L'opération API `GetGameSessionId` renvoie l'identifiant au format ARN. Type : `String` Obligatoire : oui
MatchmakingConfigurationArn	L'identifiant unique (sous la forme d'un ARN) que le système de jumelage doit utiliser pour cette demande. L'ARN du système de matchmaking correspondant à la session de jeu d'origine se trouve dans l'objet de session de jeu dans la propriété des données du système de matchmaking. Pour plus d'informations sur les données du système de jumelage, consultez la section Utilisation des données du système de jumelage. Type : `String` Obligatoire : oui
Joueurs	Ensemble de données représentant tous les joueurs qui participent actuellement à la session de jeu. Le matchmaker utilise ces informations pour rechercher de nouveaux joueurs qui constituent de bonnes correspondances pour les joueurs actuels. Type : `[]model.Player` Obligatoire : oui

Propriétés	Description
TicketId	L'identifiant unique d'un ticket de demande de matchmaking ou de remplacement de match. Si vous ne fournissez aucune valeur, Amazon en GameLift génère une. Utilisez cet identifiant pour suivre l'état du ticket de renvoi de correspondance ou annuler la requête si nécessaire. Type : `String` Obligatoire : non

Joueur

L'objet qui représente un joueur dans le matchmaking. Lorsqu'une demande de matchmaking démarre, un joueur dispose d'un identifiant de joueur, d'attributs et éventuellement de données de latence. Amazon GameLift ajoute les informations de l'équipe une fois le match terminé.

Propriétés	Description
LatencyInMS	Ensemble de valeurs exprimées en millisecondes qui indiquent le niveau de latence ressenti par un joueur lorsqu'il est connecté à un lieu. Si cette propriété est utilisée, le joueur n'est jumelé qu'aux emplacements répertoriés. Si un matchmaker dispose d'une règle qui évalue la latence, les joueurs doivent indiquer la latence pour être mis en relation. Type : `map[string] int` Obligatoire : non
PlayerAttributes	Une collection de paires clé:valeur contenant des informations sur les joueurs à utiliser dans le matchmaking. Les clés d'attribut du joueur doivent correspondre à PlayerAttributes celles utilisées dans un ensemble de règles de matchmaking. Pour plus d'informations sur les attributs des joueurs, consultez Attribute Value. Type : `map[string] AttributeValue`

Propriétés	Description
	Obligatoire : non
PlayerId	Identifiant unique pour un joueur. Type : `String` Obligatoire : non
Equipe	Le nom de l'équipe à laquelle le joueur est affecté lors d'un match. Vous définissez le nom de l'équipe dans le jeu de règles de matchmaking. Type : `String` Obligatoire : non

DescribePlayerSessionsRequest

Objet qui indique les sessions de joueur à récupérer. Le processus du serveur fournit ces informations lors d'un DescribePlayerSessions() appel à Amazon GameLift.

Propriétés	Description
`GameSessionID`	Un identifiant de session de jeu unique. Utilisez ce paramètre pour demander toutes les sessions de joueur pour la session de jeu spécifiée. Le format de l'identifiant de session de jeu est`arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string>` . `GameSessionID` Il s'agit d'une chaîne d'identification personnalisée ou d'une chaîne générée. Type : `String` Obligatoire : non
`PlayerSessionID`	Identifiant unique d'une session de joueur. Utilisez ce paramètre pour demander une session de joueur spécifique. Type : `String`

Propriétés	Description
	Obligatoire : non
`PlayerID`	L'identifiant unique d'un joueur. Utilisez ce paramètre pour demander toutes les sessions de joueur pour un joueur spécifique. veuillez consulter Générer des identifiants de joueurs. Type : `String` Obligatoire : non
`PlayerSessionStatusFilter`	État de la session du joueur sur lequel filtrer les résultats. Les statuts de session de joueur possibles incluent : • RÉSERVÉ — La demande de session du joueur a été reçue, mais le joueur ne s'est pas connecté au processus du serveur ni n'a été validé. • ACTIF — Le joueur a été validé par le processus du serveur et est connecté. • TERMINÉ — La connexion du joueur a été interrompue. • TIMEDOUT — Une demande de session de joueur a été reçue, mais le joueur ne s'est pas connecté ou n'a pas été validée dans le délai imparti (60 secondes). Type : `String` Obligatoire : non
`NextToken`	Le jeton indiquant le début de la page de résultats suivante. Pour spécifier le début du jeu de résultats, ne fournissez aucune valeur. Si vous fournissez un identifiant de session de joueur, ce paramètre est ignoré. Type : `String` Obligatoire : non

Propriétés	Description
`Limit`	Nombre maximal de résultats à renvoyer. Si vous fournissez un identifiant de session de joueur, ce paramètre est ignoré. Type : `int` Obligatoire : non

StopMatchBackfillRequest

Informations utilisées pour annuler une demande de remplissage par matchmaking. Le serveur de jeu communique ces informations au GameLift service Amazon lors d'un StopMatchBackfill() appel.

Propriétés	Description
GameSessionArn	Identifiant de session de jeu unique de la demande annulée. Type : `string` Obligatoire : non
MatchmakingConfigurationArn	L'identifiant unique du matchmaker auquel cette demande a été envoyée. Type : `string` Obligatoire : non
TicketId	L'identifiant unique du ticket de demande de remblayage à annuler. Type : `string` Obligatoire : non

GetFleetRoleCredentialsRequest

Les informations d'identification du rôle qui étendent l'accès limité à vos AWS ressources au serveur de jeu. Pour plus d’informations, consultez Configurer un rôle de service IAM pour Amazon GameLift.

Propriétés	Description
RoleArn	L'ARN du rôle de service qui étend un accès limité à vos AWS ressources. Type : `string` Obligatoire : oui
RoleSessionName	Nom de la session qui décrit l'utilisation des informations d'identification du rôle. Type : `string` Obligatoire : oui

Référence GameLift du SDK Amazon Server pour Unreal Engine

Cette référence GameLift du SDK Amazon Server peut vous aider à préparer vos projets de jeux Unreal Engine à utiliser avec Amazon. GameLift Pour plus d'informations sur le processus d'intégration, consultez Ajoutez Amazon GameLift à votre serveur de jeu.

Cette API est définie dans `GameLiftServerSDK.h` et `GameLiftServerSDKModels.h`.

Pour configurer le plug-in Unreal Engine et consulter des exemples de code Intégrer Amazon GameLift dans un projet Unreal Engine.

Rubriques

- Référence du SDK GameLift 5.x du serveur Amazon Unreal Engine
- Référence du SDK GameLift 3.x du serveur Amazon Unreal Engine

Référence du SDK GameLift 5.x du serveur Amazon Unreal Engine

Vous pouvez utiliser cette référence du SDK pour serveur Amazon GameLift Unreal Engine 5.x pour vous aider à préparer votre jeu multijoueur en vue de son utilisation avec Amazon. GameLift Pour plus de détails sur le processus d'intégration, voirAjoutez Amazon GameLift à votre serveur de jeu, et pour plus d'informations sur l'utilisation du plug-in de serveur Unreal SDK, voir. Intégrer Amazon GameLift dans un projet Unreal Engine

Rubriques

- Référence GameLift du SDK Amazon Server (Unreal) 5.x : Actions
- Référence GameLift du SDK Amazon Server (Unreal) : types de données

Référence GameLift du SDK Amazon Server (Unreal) 5.x : Actions

Vous pouvez utiliser cette référence du SDK Amazon GameLift Unreal Server pour vous aider à préparer votre jeu multijoueur en vue de son utilisation avec Amazon. GameLift Pour plus de détails sur le processus d'intégration, voir Ajoutez Amazon GameLift à votre serveur de jeu et pour plus d'informations sur l'utilisation du plug-in de serveur Unreal SDK, voir. Intégrer Amazon GameLift dans un projet Unreal Engine

Actions

- GetSdkVersion()
- InitSDK()
- InitSDK()
- ProcessReady()
- ProcessEnding()
- ActivateGameSession()
- UpdatePlayerSessionCreationPolicy()
- GetGameSessionId()
- GetTerminationTime()
- AcceptPlayerSession()
- RemovePlayerSession()
- DescribePlayerSessions()
- StartMatchBackfill()
- StopMatchBackfill()
- GetComputeCertificate()
- GetFleetRoleCredentials()

> **Note**
>
> Cette rubrique décrit l'API Amazon GameLift C++ que vous pouvez utiliser lorsque vous créez pour Unreal Engine. Plus précisément, cette documentation s'applique au code que vous compilez avec l'`-DBUILD_FOR_UNREAL=1`option.

GetSdkVersion()

Renvoie le numéro de version actuel du kit SDK intégré dans le processus serveur.

Syntaxe

```
FGameLiftStringOutcome GetSdkVersion();
```

Valeur renvoyée

En cas de réussite, renvoie la version actuelle du kit SDK en tant qu'objet the section called “F GameLiftStringOutcome”. L'objet renvoyé inclut le numéro de version (exemple`5.0.0`). En cas d'échec, renvoie un message d'erreur.

Exemple

```
Aws::GameLift::AwsStringOutcome SdkVersionOutcome =
 Aws::GameLift::Server::GetSdkVersion();
```

InitSDK()

Initialise le GameLift SDK Amazon pour un parc EC2 géré. Appelez cette méthode au lancement, avant toute autre initialisation liée à Amazon GameLift . Cette méthode lit les paramètres du serveur depuis l'environnement hôte afin de configurer la communication entre le serveur et le GameLift service Amazon.

Syntaxe

```
FGameLiftGenericOutcome InitSDK()
```

Valeur renvoyée

En cas de succès, renvoie un `InitSdkOutcome` objet indiquant que le processus serveur est prêt à être appeléProcessReady().

Exemple

```
//Call InitSDK to establish a local connection with the GameLift agent to enable
 further communication.
FGameLiftGenericOutcome initSdkOutcome = gameLiftSdkModule->InitSDK();
```

InitSDK()

Initialise le GameLift SDK Amazon pour une Anywhere flotte. Appelez cette méthode au lancement, avant toute autre initialisation liée à Amazon GameLift . Cette méthode nécessite des paramètres de serveur explicites pour configurer la communication entre le serveur et le GameLift service Amazon.

Syntaxe

```
FGameLiftGenericOutcome InitSDK(serverParameters)
```

Paramètres

F ServerParameters

Pour initialiser un serveur de jeu sur une GameLift Anywhere flotte Amazon, créez un `ServerParameters` objet avec les informations suivantes :

- URL de l'URL WebSocket utilisée pour se connecter à votre serveur de jeu.
- ID du processus utilisé pour héberger votre serveur de jeu.
- L'ID de l'ordinateur hébergeant les processus de votre serveur de jeu.
- L'ID de la GameLift flotte Amazon contenant votre GameLift Anywhere ordinateur Amazon.
- Le jeton d'autorisation généré par l' GameLift opération Amazon.

Valeur renvoyée

En cas de succès, renvoie un `InitSdkOutcome` objet indiquant que le processus serveur est prêt à être appeléProcessReady().

> **Note**
>
> Si les appels à échouent pour `InitSDK()` les builds de jeu déployés sur des flottes Anywhere, vérifiez le `ServerSdkVersion` paramètre utilisé lors de la création de la ressource de build. Vous devez définir explicitement cette valeur en fonction de la version du SDK du serveur utilisée. La valeur par défaut de ce paramètre est 4.x, ce qui n'est pas

compatible. Pour résoudre ce problème, créez une nouvelle version et déployez-la sur une nouvelle flotte.

Exemple

```
//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()

Indique à Amazon GameLift que le processus du serveur est prêt à héberger des sessions de jeu. Appelez cette méthode après l'avoir invoquée. InitSDK() Cette méthode ne doit être appelée qu'une seule fois par processus.

Syntaxe

```
GenericOutcome ProcessReady(const Aws::GameLift::Server::ProcessParameters
&processParameters);
```

Paramètres

processParameters

F ProcessParametersObjet communiquant les informations suivantes concernant le processus du serveur :

- Noms des méthodes de rappel implémentées dans le code du serveur de jeu invoqué par le GameLift service Amazon pour communiquer avec le processus du serveur.
- Numéro de port sur lequel le processus serveur écoute.
- Chemin d'accès à tous les fichiers spécifiques à une session de jeu que vous souhaitez qu'Amazon capture et GameLift stocke.

Valeur renvoyée

Renvoie un résultat générique consistant en un succès ou un échec avec un message d'erreur.

Exemple

Cet exemple illustre les implémentations de l'appel ProcessReady() et de la fonction déléguée.

```
//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()

Indique à Amazon GameLift que le processus du serveur est en train de se terminer. Appelez cette méthode après toutes les autres tâches de nettoyage (y compris la fermeture de la session de jeu active) et avant de terminer le processus. En fonction du résultat de`ProcessEnding()`, le processus se termine avec succès (0) ou erreur (-1) et génère un événement de flotte. Si le processus se termine par une erreur, l'événement de flotte généré est`SERVER_PROCESS_TERMINATED_UNHEALTHY`).

Syntaxe

```
FGameLiftGenericOutcome ProcessEnding()
```

Valeur renvoyée

Renvoie un résultat générique consistant en un succès ou un échec avec un message d'erreur.

Exemple

```
//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()

Informe Amazon GameLift que le processus du serveur a activé une session de jeu et qu'il est désormais prêt à recevoir les connexions des joueurs. Cette action doit être appelée dans le cadre de la fonction de `onStartGameSession()` rappel, après toute initialisation de session de jeu.

Syntaxe

```
FGameLiftGenericOutcome ActivateGameSession()
```

Valeur renvoyée

Renvoie un résultat générique consistant en un succès ou un échec avec un message d'erreur.

Exemple

Cet exemple montre l'`ActivateGameSession()`appel dans le cadre de la fonction de `onStartGameSession()` délégation.

```
//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()

Met à jour la capacité de la session de jeu à accepter de nouvelles sessions de joueur. Une session de jeu peut être définie pour accepter ou refuser toutes les nouvelles sessions joueur.

Syntaxe

```
FGameLiftGenericOutcome UpdatePlayerSessionCreationPolicy(EPlayerSessionCreationPolicy
 policy)
```

Paramètres

playerCreationSessionPolitique

Valeur de chaîne indiquant si la session de jeu accepte ou non de nouveaux joueurs.

Les valeurs valides sont les suivantes :

- ACCEPT_ALL — Accepte toutes les sessions de nouveaux joueurs.
- DENY_ALL — Refuse toutes les sessions de nouveaux joueurs.

Valeur renvoyée

Renvoie un résultat générique consistant en un succès ou un échec avec un message d'erreur.

Exemple

Cet exemple définit la stratégie de participation de la session de jeu actuelle de manière à ce que tous les joueurs soient acceptés.

```
FGameLiftGenericOutcome outcome = gameLiftSdkModule-
>UpdatePlayerSessionCreationPolicy(Aws::GameLift::Model::EPlayerSessionCreationPolicy::ACCEPT_A
```

GetGameSessionId()

Récupère l'ID de la session de jeu hébergée par le processus serveur actif.

Pour les processus inactifs qui ne sont pas activés lors d'une session de jeu, l'appel renvoie unthe section called “F GameLiftError”.

Syntaxe

```
FGameLiftStringOutcome GetGameSessionId()
```

Paramètres

Cette action n'a aucun paramètre.

Valeur renvoyée

En cas de réussite, renvoie l'ID de session de jeu en tant qu'objet the section called “F GameLiftStringOutcome”. En cas d'échec, renvoie un message d'erreur. »

Pour les processus inactifs qui ne sont pas activés lors d'une session de jeu, l'appel renvoie `Success = True` et `GameSessionId =""`.

Exemple

```
//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()

Renvoie l'heure d'arrêt planifiée pour un processus serveur, si une heure de résiliation est disponible. Un processus serveur prend des mesures après avoir reçu un `onProcessTerminate()` rappel d'Amazon GameLift. Amazon GameLift appelle `onProcessTerminate()` pour les raisons suivantes :

- Lorsque le processus du serveur a signalé un mauvais état de santé ou n'a pas répondu à Amazon GameLift.
- Lorsque vous mettez fin à l'instance lors d'un événement de réduction de la taille.
- Lorsqu'une instance est interrompue en raison d'une interruption ponctuelle.

Syntaxe

```
AwsDateTimeOutcome GetTerminationTime()
```

Valeur renvoyée

En cas de succès, renvoie l'heure de fin sous forme d'`AwsDateTimeOutcome`objet. La valeur est le délai de résiliation, exprimé en ticks écoulés depuis. `0001 00:00:00` Par exemple, la valeur de la date et de l'heure `2020-09-13 12:26:40 -000Z` est égale à celle `637355968000000000` des ticks. Si aucune heure de résiliation n'est disponible, renvoie un message d'erreur.

Si le processus n'a pas reçu de `ProcessParameters.OnProcessTerminate()` rappel, un message d'erreur est renvoyé. Pour plus d'informations sur l'arrêt d'un processus serveur, consultezRépondre à une notification d'arrêt d'un processus serveur.

Exemple

```
AwsDateTimeOutcome TermTimeOutcome = gameLiftSdkModule->GetTerminationTime();
```

AcceptPlayerSession()

Informe Amazon GameLift qu'un joueur possédant l'identifiant de session de joueur spécifié s'est connecté au processus du serveur et doit être validé. Amazon GameLift vérifie que l'identifiant de session du joueur est valide. Une fois la session du joueur validée, Amazon GameLift change le statut de l'emplacement du joueur de RÉSERVÉ à ACTIF.

Syntaxe

```
FGameLiftGenericOutcome AcceptPlayerSession(const FString& playerSessionId)
```

Paramètres

playerSessionId

Identifiant unique émis par Amazon GameLift lors de la création d'une nouvelle session de joueur.

Valeur renvoyée

Renvoie un résultat générique consistant en un succès ou un échec avec un message d'erreur.

Exemple

Cet exemple traite une demande de connexion qui inclut la validation et le rejet d'identifiants de session de joueur non valides.

```
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()

Indique à Amazon GameLift qu'un joueur s'est déconnecté du processus du serveur. En réponse, Amazon GameLift modifie l'emplacement du joueur pour le rendre disponible.

Syntaxe

```
FGameLiftGenericOutcome RemovePlayerSession(const FString& playerSessionId)
```

Paramètres

`playerSessionId`

Identifiant unique émis par Amazon GameLift lors de la création d'une nouvelle session de joueur.

Valeur renvoyée

Renvoie un résultat générique consistant en un succès ou un échec avec un message d'erreur.

Exemple

```
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()

Récupère les données de session du joueur, notamment les paramètres, les métadonnées de session et les données du joueur. Utilisez cette méthode pour obtenir des informations sur les points suivants :

- Une session solo
- Toutes les sessions des joueurs au cours d'une session de jeu
- Toutes les sessions de joueur associées à un identifiant de joueur unique

Syntaxe

```
FGameLiftDescribePlayerSessionsOutcome DescribePlayerSessions(const
 FGameLiftDescribePlayerSessionsRequest &describePlayerSessionsRequest)
```

Paramètres

F GameLiftDescribePlayerSessionsRequest

the section called “F GameLiftDescribePlayerSessionsRequest”Objet qui décrit les sessions de joueur à récupérer.

Valeur renvoyée

En cas de réussite, renvoie un objet the section called “F GameLiftDescribePlayerSessionsOutcome” qui contient un ensemble d'objets de session de joueur correspondant aux paramètres de la demande.

Exemple

Cet exemple demande toutes les sessions de joueur activement connectées à une session de jeu spécifiée. En omettant NextTokenet en définissant la valeur limite sur 10, Amazon GameLift renvoie les 10 premiers enregistrements de session de joueur correspondant à la demande.

```
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()

Envoie une demande de recherche de nouveaux joueurs pour des emplacements ouverts dans une session de jeu créée avec FlexMatch. Pour plus d'informations, voir la fonction de FlexMatch remblayage.

Cette action est asynchrone. Si de nouveaux joueurs sont jumelés, Amazon GameLift fournit des données de matchmaking mises à jour à l'aide de la fonction de rappel. `OnUpdateGameSession()`

Un processus de serveur ne peut comporter qu'une seule requête de renvoi de correspondance à la fois. Pour envoyer une nouvelle requête, appelez d'abord StopMatchBackfill() pour annuler la requête d'origine.

Syntaxe

```
FGameLiftStringOutcome StartMatchBackfill (FStartMatchBackfillRequest
 &startBackfillRequest);
```

Paramètres

F StartMatchBackfillRequest

Un StartMatchBackfillRequest objet qui communique les informations suivantes :

- ID de ticket à attribuer à la requête de renvoi. Ces informations sont facultatives ; si aucun identifiant n'est fourni, Amazon en GameLift générera un.
- Matchmaker auquel envoyer la requête. L'ARN de configuration complet est obligatoire. Cette valeur se trouve dans les données du matchmaker de la session de jeu.
- ID de la session de jeu à compléter.
- Les données de matchmaking disponibles pour les joueurs actuels de la session de jeu.

Valeur renvoyée

Renvoie un `StartMatchBackfillOutcome` objet avec l'identifiant du ticket de remplacement correspondant, ou un échec avec un message d'erreur.

Exemple

```
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()

Annule une demande de remplacement de match active. Pour plus d'informations, voir la fonction de FlexMatch remblayage.

Syntaxe

```
FGameLiftGenericOutcome StopMatchBackfill (FStopMatchBackfillRequest
 &stopBackfillRequest);
```

Paramètres

F StopMatchBackfillRequest

Un StopMatchBackfillRequest objet identifiant le ticket de matchmaking à annuler :

- L'identifiant du ticket attribué à la demande de remblayage.
- L'entremetteur à qui la demande de remblayage a été envoyée.
- La session de jeu associée à la demande de remplacement.

Valeur renvoyée

Renvoie un résultat générique consistant en un succès ou un échec avec un message d'erreur.

Exemple

```
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()

Récupère le chemin d'accès au certificat TLS utilisé pour chiffrer la connexion réseau entre votre ressource GameLift Anywhere informatique Amazon et Amazon. GameLift Vous pouvez utiliser le chemin du certificat lorsque vous enregistrez votre appareil informatique dans une GameLift Anywhere flotte Amazon. Pour plus d'informations, voir, RegisterCompute.

Syntaxe

```
FGameLiftGetComputeCertificateOutcome FGameLiftServerSDKModule::GetComputeCertificate()
```

Valeur renvoyée

Renvoie un `GetComputeCertificateResponse` objet contenant les éléments suivants :

- CertificatePath: chemin d'accès au certificat TLS sur votre ressource de calcul.
- HostName: nom d'hôte de votre ressource de calcul.

Exemple

```
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()

Récupère les informations d'identification du rôle IAM qui autorisent Amazon GameLift à interagir avec d'autres. Services AWS Pour plus d'informations, consultez Communiquez avec les autres AWS ressources de vos flottes.

Syntaxe

```
FGameLiftGetFleetRoleCredentialsOutcome
 FGameLiftServerSDKModule::GetFleetRoleCredentials(const
 FGameLiftGetFleetRoleCredentialsRequest &request)
```

Paramètres

F GameLiftGetFleetRoleCredentialsRequest

Valeur renvoyée

Renvoie un objet the section called "F GameLiftGetFleetRoleCredentialsOutcome".

Exemple

```
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
}
```

Référence GameLift du SDK Amazon Server (Unreal) : types de données

Vous pouvez utiliser cette référence du SDK du serveur Amazon GameLift Unreal pour vous aider à préparer votre jeu multijoueur en vue de son utilisation avec Amazon. GameLift Pour plus de détails sur le processus d'intégration, voir Ajoutez Amazon GameLift à votre serveur de jeu et pour plus d'informations sur l'utilisation du plug-in de serveur Unreal SDK, voir. Intégrer Amazon GameLift dans un projet Unreal Engine

Types de données

- 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
- Enums

> **Note**
>
> Cette rubrique décrit l'API Amazon GameLift C++ que vous pouvez utiliser lorsque vous créez pour Unreal Engine. Plus précisément, cette documentation s'applique au code que vous compilez avec l'`-DBUILD_FOR_UNREAL=1`option.

F ProcessParameters

Ce type de données contient l'ensemble de paramètres envoyé à Amazon GameLift dans un fichierProcessReady().

Propriétés	Description

LogParameters	Objet avec des chemins de répertoire vers des fichiers générés au cours d'une session de jeu. Amazon GameLift copie et stocke les fichiers pour un accès futur. Type : `TArray<FString>` Obligatoire : non
OnHealthCheck	Fonction de rappel GameLift invoquée par Amazon pour demander un rapport d'état de santé au processus du serveur. Amazon GameLift appelle cette fonction toutes les 60 secondes et attend une réponse pendant 60 secondes. Le processus du serveur revient `TRUE` s'il est sain, `FALSE` sinon sain. Si aucune réponse n'est renvoyée, Amazon considère GameLift que le processus du serveur n'est pas sain. Cette propriété est une fonction déléguée définie comme suit `DECLARE_DELEGATE_RetVal(bool, FOnHealthCheck)` : Type : `FOnHealthCheck` Obligatoire : non
OnProcessTerminate	La fonction de rappel GameLift invoquée par Amazon pour forcer le processus du serveur à s'arrêter. Après avoir appelé cette fonction, Amazon GameLift attend 5 minutes que le processus du serveur s'arrête et répond par un ProcessEnding() appel avant d'arrêter le processus du serveur. Type : `FSimpleDelegate` Obligatoire : oui

OnStartGameSession	La fonction de rappel GameLift invoquée par Amazon pour activer une nouvelle session de jeu. Amazon GameLift appelle cette fonction en réponse à une demande d'un client CreateGameSession. La fonction de rappel transmet un GameSessionobjet, tel que défini dans le Amazon GameLift API Reference. Cette propriété est une fonction déléguée définie comme `DECLARE_DELEGATE_OneParam(FOnStartGameSession, Aws::GameLift::Server::Model::GameSession);` Type : `FOnStartGameSession` Obligatoire : oui
OnUpdateGameSession	Fonction de rappel GameLift invoquée par Amazon pour transmettre un objet de session de jeu mis à jour au processus du serveur. Amazon GameLift appelle cette fonction lorsqu'une demande de mise en correspondance a été traitée pour fournir des données de matchmaking mises à jour. Il transmet un GameSessionobjet, une mise à jour de statut (`updateReason`) et l'identifiant du ticket de remplacement des matchs. Cette propriété est une fonction déléguée définie comme `DECLARE_DELEGATE_OneParam(FOnUpdateGameSession, Aws::GameLift::Server::Model::UpdateGameSession);` Type : `FOnUpdateGameSession` Obligatoire : non

Port	Numéro de port sur lequel le processus serveur écoute les connexions des nouveaux joueurs. La valeur doit être comprise dans la plage de ports configurée pour toutes les flottes déployant cette version de génération du serveur de jeux. Ce numéro de port est inclus dans les objets de session de jeu et de session de joueur, que les sessions de jeu utilisent pour se connecter à un processus serveur. Type : `int` Obligatoire : oui

UpdateGameSession

Ce type de données est mis à jour vers un objet de session de jeu, qui inclut la raison pour laquelle la session de jeu a été mise à jour et l'identifiant du ticket de remplissage associé si le remplissage est utilisé pour remplir les sessions des joueurs pendant la session de jeu.

Propriétés	Description
GameSession	Un GameSessionobjet défini par l' GameLift API Amazon. L'`GameSession` objet contient des propriétés décrivant une session de jeu. Type : `Aws::GameLift::Server::GameSession` Obligatoire : non
UpdateReason	La raison pour laquelle la session de jeu est mise à jour. Type : `enum class UpdateReason` • DONNÉES DE MATCHMAKING MISES À JOUR

Propriétés	Description
	• BACKFILL_FAILED • BACKFILL_TIMED_OUT • BACKFILL_ANNULÉ Obligatoire : non
BackfillTicketId	L'identifiant du ticket de remplacement qui tente de mettre à jour la session de jeu. Type : `char[]` Obligatoire : non

GameSession

Ce type de données fournit les détails d'une session de jeu.

Propriétés	Description
GameSessionId	Identifiant unique pour la session de jeu. L'ARN d'une session de jeu a le format suivant :`arn:aws:gamelift:<region>::gamesession/<fleet ID>/<custom ID string or idempotency token>` . Type : `char[]` Obligatoire : non
Nom	Une étiquette descriptive de la session de jeu. Type : `char[]` Obligatoire : non
FleetId	Identifiant unique de la flotte sur laquelle s'exécute la session de jeu.

Propriétés	Description
	Type : `char[]` Obligatoire : non
MaximumPlayerSessionCount	Le nombre maximum de connexions de joueurs à la session de jeu. Type : `int` Obligatoire : non
Port	Le numéro de port de la session de jeu. Pour se connecter à un serveur de GameLift jeu Amazon, une application a besoin à la fois de l'adresse IP et du numéro de port. Type : `int` Obligatoire : non
IpAddress	Adresse IP de la session de jeu. Pour se connecter à un serveur de GameLift jeu Amazon, une application a besoin à la fois de l'adresse IP et du numéro de port. Type : `char[]` Obligatoire : non
GameSessionData	Ensemble de propriétés de session de jeu personnalisées, mises en forme en tant que valeur de chaîne unique. Type : `char[]` Obligatoire : non

Propriétés	Description
MatchmakerData	Informations sur le processus de matchmaking utilisé pour créer la session de jeu, en syntaxe JSON, formatée sous forme de chaîne. En plus de la configuration de matchmaking utilisée, il contient des données sur tous les joueurs affectés au match, y compris les attributs des joueurs et les affectations des équipes. Type : `char[]` Obligatoire : non
GameProperties	Ensemble de propriétés personnalisées pour une session de jeu, formatées sous forme de paires clé:valeur. Ces propriétés sont transmises avec une demande de démarrage d'une nouvelle session de jeu. Type : `GameProperty[]` Obligatoire : non

Propriétés	Description
DnsName	Identifiant DNS attribué à l'instance qui exécute la session de jeu. Les valeurs ont le format suivant : • Flottes compatibles TLS :. `<unique identifier>.<region identifier>.amazongamelift.com` • Flottes non compatibles TLS :. `ec2-<unique identifier>.compute.amazonaws.com` Lorsque vous vous connectez à une session de jeu exécutée sur une flotte compatible TLS, vous devez utiliser le nom DNS et non l'adresse IP. Type : `char[]` Obligatoire : non

F ServerParameters

Informations utilisées pour maintenir la connexion entre un GameLift Anywhere serveur Amazon et le GameLift service Amazon. Ces informations sont utilisées lors du lancement de nouveaux processus serveur avecInitSDK(). Pour les serveurs hébergés sur des instances EC2 GameLift gérées par Amazon, utilisez un objet vide.

Propriétés	Description
webSocketUrl	L'`GameLiftServerSdkEndpoint` Amazon GameLift revient lorsque vous `RegisterCompute` recherchez une ressource GameLift Anywhere informatique Amazon. Type : `char[]`

Propriétés	Description
	Obligatoire : oui
ID du processus	Un identifiant unique enregistré auprès du processus serveur hébergeant votre jeu. Type : `char[]` Obligatoire : oui
hostId	Le `HostID` est celui `ComputeName` utilisé lorsque vous avez enregistré votre ordinateur. Pour plus d'informations, voir, RegisterCompute. Type : `char[]` Obligatoire : oui
ID de flotte	Identifiant unique de la flotte dans laquelle le calcul est enregistré. Pour plus d'informations, voir, RegisterCompute. Type : `char[]` Obligatoire : oui
Jeton d'authentification	Le jeton d'authentification généré par Amazon GameLift qui authentifie votre serveur auprès d'Amazon GameLift. Pour plus d'informations, voir, GetComputeAuthToken. Type : `char[]` Obligatoire : oui

F StartMatchBackfillRequest

Informations utilisées pour créer une demande de remplissage par matchmaking. Le serveur de jeu communique ces informations à Amazon GameLift lors d'un StartMatchBackfill() appel.

Propriétés	Description
GameSessionArn	Un identifiant de session de jeu unique. L'opération API `GetGameSessionId` renvoie l'identifiant au format ARN. Type : `char[]` Obligatoire : oui
MatchmakingConfigurationArn	Un identifiant unique, sous la forme d'un ARN, que le système de jumelage doit utiliser pour cette demande. L'ARN du système de matchmaking correspondant à la session de jeu d'origine se trouve dans l'objet de session de jeu dans la propriété des données du système de matchmaking. Pour en savoir plus sur les données du système de jumelage, consultez la section Travailler avec les données du système de jumelage. Type : `char[]` Obligatoire : oui
Joueurs	Ensemble de données représentant tous les joueurs participant à la session de jeu. Le matchmaker utilise ces informations pour rechercher de nouveaux joueurs qui constitue nt de bonnes correspondances pour les joueurs actuels. Type : `TArray<FPlayer>`

Propriétés	Description
	Obligatoire : oui
TicketId	Un identifiant unique pour un ticket de demande de matchmaking ou de remplacement de match. Si vous ne fournissez aucune valeur, Amazon en GameLift génère une. Utilisez cet identifiant pour suivre l'état du ticket de renvoi de correspondance ou annuler la requête si nécessaire. Type : `char[]` Obligatoire : non

FPlayer

Ce type de données représente un joueur dans le matchmaking. Lors du lancement d'une demande de matchmaking, un joueur dispose d'un identifiant de joueur, d'attributs et éventuellement de données de latence. Amazon GameLift ajoute les informations de l'équipe une fois le match terminé.

Propriétés	Description
LatencyInMS	Ensemble de valeurs exprimées en milliseco ndes qui indiquent le niveau de latence ressenti par un joueur lorsqu'il est connecté à un lieu. Si cette propriété est utilisée, le joueur n'est jumelé qu'aux emplacements répertoriés. Si un matchmaker dispose d'une règle qui évalue la latence, les joueurs doivent indiquer la latence pour être mis en relation. Type : `TMap>FString, int32<` Obligatoire : non

Propriétés	Description
PlayerAttributes	Une collection de paires clé:valeur contenant des informations sur les joueurs à utiliser dans le matchmaking. Les clés d'attribut du joueur doivent correspondre à PlayerAttributes celles utilisées dans un ensemble de règles de matchmaking. Pour plus d'informations sur les attributs des joueurs, consultez AttributeValue. Type : `TMap>FString, FAttribut eValue<` Obligatoire : non
PlayerId	Identifiant unique pour un joueur. Type : `std::string` Obligatoire : non
Equipe	Le nom de l'équipe à laquelle le joueur est affecté lors d'un match. Vous définissez le nom de l'équipe dans le jeu de règles de matchmaki ng. Type : `FString` Obligatoire : non

F GameLiftDescribePlayerSessionsRequest

Objet qui indique les sessions de joueur à récupérer. Le processus du serveur fournit ces informations lors d'un DescribePlayerSessions() appel à Amazon GameLift.

Propriétés	Description
GameSessionId	Un identifiant de session de jeu unique. Utilisez ce paramètre pour demander toutes les sessions de joueur pour la session de jeu spécifiée. Le format de l'identifiant de session de jeu estFString. GameSessionID Il s'agit d'une chaîne d'identification personnalisée ou d'un Type : std::string Obligatoire : non
PlayerSessionId	Identifiant unique d'une session de joueur. Utilisez ce paramètre pour demander une session de joueur spécifique. Type : FString Obligatoire : non
PlayerId	L'identifiant unique d'un joueur. Utilisez ce paramètre pour demander toutes les sessions de joueur pour un joueur spécifique. veuillez consulter Générer des identifiants de joueurs. Type : FString Obligatoire : non
PlayerSessionStatusFilter	État de la session du joueur sur lequel filtrer les résultats. Les statuts possibles des sessions de joueur sont les suivants : • RÉSERVÉ — La demande de session du joueur a été reçue, mais le joueur ne s'est pas connecté au processus du serveur ni n'a été validé.

Propriétés	Description
	• ACTIF — Le joueur a été validé par le processus du serveur et est connecté. • TERMINÉ — La connexion du joueur a été interrompue. • TIMEDOUT — Une demande de session de joueur a été reçue, mais le joueur ne s'est pas connecté ou n'a pas été validée dans le délai imparti (60 secondes). Type : `FString` Obligatoire : non
NextToken	Le jeton indiquant le début de la page de résultats suivante. Pour spécifier le début du jeu de résultats, ne fournissez aucune valeur. Si vous fournissez un identifiant de session de joueur, ce paramètre est ignoré. Type : `FString` Obligatoire : non
Limite	Nombre maximal de résultats à renvoyer. Si vous fournissez un identifiant de session de joueur, ce paramètre est ignoré. Type : `int` Obligatoire : non

F StopMatchBackfillRequest

Informations utilisées pour annuler une demande de remplissage par matchmaking. Le serveur de jeu communique ces informations au GameLift service Amazon lors d'un StopMatchBackfill() appel.

Propriétés	Description
GameSessionArn	Identifiant de session de jeu unique de la demande annulée. Type : `FString` Obligatoire : oui
MatchmakingConfigurationArn	Un identifiant unique du matchmaker auquel cette demande a été envoyée. Type : `FString` Obligatoire : oui
TicketId	Identifiant unique du ticket de demande de remblayage à annuler. Type : `FString` Obligatoire : oui

F AttributeValue

Utilisez ces valeurs dans les paires FPlayer clé-valeur d'attribut. Cet objet vous permet de spécifier une valeur d'attribut à l'aide de n'importe quel type de données valide : chaîne, nombre, tableau de chaînes ou mappage de données. Chaque `AttributeValue` objet ne peut utiliser qu'une seule des propriétés disponibles.

Propriétés	Description
Type ATR	Spécifie le type de valeur d'attribut. Type : valeur `FAttributeType` enum. Obligatoire : non
S	Représente une valeur d'attribut de chaîne.

Propriétés	Description
	Type : `FString` Obligatoire : non
N	Représente une valeur d'attribut numérique. Type : `double` Obligatoire : non
SL	Représente un tableau de valeurs d'attributs de chaîne. Type : `TArray<FString>` Obligatoire : non
SDM	Représente un dictionnaire de clés de chaîne et de valeurs doubles. Type : `TMap<FString, double>` Obligatoire : non

F GameLiftGetFleetRoleCredentialsRequest

Ce type de données fournit des informations d'identification de rôle qui étendent l'accès limité à vos AWS ressources au serveur de jeu. Pour plus d'informations, consultez Configurer un rôle de service IAM pour Amazon GameLift.

Propriétés	Description
RoleArn	Le nom de ressource Amazon (ARN) du rôle de service qui étend un accès limité à vos AWS ressources. Type : `FString`

Propriétés	Description
	Obligatoire : non
RoleSessionName	Nom de la session décrivant l'utilisation des informations d'identification du rôle. Type : `FString` Obligatoire : non

F GameLiftLongOutcome

Ce type de données résulte d'une action et produit un objet doté des propriétés suivantes :

Propriétés	Description
Résultat	Le résultat de l'action. Type : `long` Obligatoire : non
ResultWithOwnership	Le résultat de l'action, converti en valeur rvalue, afin que le code appelant puisse s'approprier l'objet. Type : `long&&` Obligatoire : non
Réussite	Que l'action ait été couronnée de succès ou non. Type : `bool` Obligatoire : oui
Erreur	Erreur survenue en cas d'échec de l'action. Type : the section called “F GameLiftError”

Propriétés	Description
	Obligatoire : non

F GameLiftStringOutcome

Ce type de données résulte d'une action et produit un objet doté des propriétés suivantes :

Propriétés	Description
Résultat	Le résultat de l'action. Type : `FString` Obligatoire : non
ResultWithOwnership	Le résultat de l'action, converti en valeur rvalue, afin que le code appelant puisse s'approprier l'objet. Type : `FString&&` Obligatoire : non
Réussite	Que l'action ait été couronnée de succès ou non. Type : `bool` Obligatoire : oui
Erreur	Erreur survenue en cas d'échec de l'action. Type : the section called “F GameLiftError” Obligatoire : non

F GameLiftDescribePlayerSessionsOutcome

Ce type de données résulte d'une action et produit un objet doté des propriétés suivantes :

Propriétés	Description
Résultat	Le résultat de l'action. Type : the section called “F GameLiftDescribePlayerSessionsResult” Obligatoire : non
ResultWithOwnership	Le résultat de l'action, converti en valeur rvalue, afin que le code appelant puisse s'approprier l'objet. Type : `FGameLiftDescribePlayerSessionsResult&&` Obligatoire : non
Réussite	Que l'action ait été couronnée de succès ou non. Type : `bool` Obligatoire : oui
Erreur	Erreur survenue en cas d'échec de l'action. Type : the section called “F GameLiftError” Obligatoire : non

F GameLiftDescribePlayerSessionsResult

Propriétés	Description
PlayerSessions	Type : `TArray<FGameLiftPlayerSession>`

Propriétés	Description
	Obligatoire : oui
NextToken	Le jeton indiquant le début de la page de résultats suivante. Pour spécifier le début du jeu de résultats, ne fournissez aucune valeur. Si vous fournissez un identifiant de session de joueur, ce paramètre est ignoré. Type : `FString` Obligatoire : non
Réussite	Que l'action ait été couronnée de succès ou non. Type : `bool` Obligatoire : oui
Erreur	Erreur survenue en cas d'échec de l'action. Type : the section called “F GameLiftError” Obligatoire : non

F GenericOutcome

Ce type de données résulte d'une action et produit un objet doté des propriétés suivantes :

Propriétés	Description
Réussite	Que l'action ait été couronnée de succès ou non. Type : `bool` Obligatoire : oui
Erreur	Erreur survenue en cas d'échec de l'action.

Propriétés	Description
	Type : the section called “F GameLiftError” Obligatoire : non

F GameLiftPlayerSession

Propriétés	Description
CreationTime	Type : `long` Obligatoire : oui
FleetId	Type : `FString` Obligatoire : oui
GameSessionId	Type : `FString` Obligatoire : oui
IpAddress	Type : `FString` Obligatoire : oui
PlayerData	Type : `FString` Obligatoire : oui
PlayerId	Type : `FString` Obligatoire : oui
PlayerSessionId	Type : `FString` Obligatoire : oui
Port	Type : `int` Obligatoire : oui

Propriétés	Description
Statut	Type : Une `PlayerSessionStatus` énumération. Obligatoire : oui
TerminationTime	Type : `long` Obligatoire : oui
DnsName	Type : `FString` Obligatoire : oui

F GameLiftGetComputeCertificateOutcome

Ce type de données résulte d'une action et produit un objet doté des propriétés suivantes :

Propriétés	Description
Résultat	Le résultat de l'action. Type : the section called “F GameLiftGetComputeCertificateResult” Obligatoire : non
ResultWithOwnership	Le résultat de l'action, converti en valeur rvalue, afin que le code appelant puisse s'approprier l'objet. Type : `FGameLiftGetComputeCertificateResult&&` Obligatoire : non
Réussite	Que l'action ait été couronnée de succès ou non.

Propriétés	Description
	Type : `bool` Obligatoire : oui
Erreur	Erreur survenue en cas d'échec de l'action. Type : the section called “F GameLiftError” Obligatoire : non

F GameLiftGetComputeCertificateResult

Le chemin d'accès au certificat TLS sur votre ordinateur et le nom d'hôte du calcul.

Propriétés	Description
CertificatePath	Type : `FString` Obligatoire : oui
ComputeName	Type : `FString` Obligatoire : oui

F GameLiftGetFleetRoleCredentialsOutcome

Ce type de données résulte d'une action et produit un objet doté des propriétés suivantes :

Propriétés	Description
Résultat	Le résultat de l'action. Type : the section called “F GetFleetRoleCredentialsResult” Obligatoire : non

Propriétés	Description
ResultWithOwnership	Le résultat de l'action, converti en valeur rvalue, afin que le code appelant puisse s'approprier l'objet. Type : `FGameLiftGetFleetRoleCredentialsResult&&` Obligatoire : non
Réussite	Que l'action ait été couronnée de succès ou non. Type : `bool` Obligatoire : oui
Erreur	Erreur survenue en cas d'échec de l'action. Type : the section called “F GameLiftError” Obligatoire : non

F GetFleetRoleCredentialsResult

Propriétés	Description
AccessKeyId	ID de clé d'accès permettant d'authentifier et de fournir un accès à vos AWS ressources. Type : `FString` Obligatoire : non
AssumedRoleId	ID de l'utilisateur auquel appartient le rôle de service. Type : `FString`

Propriétés	Description
	Obligatoire : non
AssumedRoleUserArn	Le nom de ressource Amazon (ARN) de l'utilisateur auquel appartient le rôle de service. Type : `FString` Obligatoire : non
Expiration	Durée avant l'expiration de vos informations d'identification de session. Type : `FDateTime` Obligatoire : non
SecretAccessKey	ID de clé d'accès secrète pour l'authentification. Type : `FString` Obligatoire : non
SessionToken	Un jeton permettant d'identifier la session active en cours qui interagit avec vos AWS ressources. Type : `FString` Obligatoire : non
Réussite	Que l'action ait été couronnée de succès ou non. Type : `bool` Obligatoire : oui

Propriétés	Description
Erreur	Erreur survenue en cas d'échec de l'action. Type : the section called “GameLiftError” Obligatoire : non

F GameLiftError

Propriétés	Description
ErrorType	Type d'erreur. Type : Une `GameLiftErrorType` énumération. Obligatoire : non
ErrorName	Nom de l'erreur. Type : `std::string` Obligatoire : non
ErrorMessage	Message d’erreur. Type : `std::string` Obligatoire : non

Enums

Les énumérations définies pour le SDK GameLift du serveur Amazon (Unreal) sont définies comme suit :

F AttributeType

- NONE
- CHAÎNE

- DOUBLE
- LISTE_CHAÎNES
- STRING_DOUBLE_MAP

GameLiftErrorType

Valeur de chaîne indiquant le type d'erreur. Les valeurs valides sont les suivantes :

- SERVICE_CALL_FAILED — Un appel à un AWS service a échoué.
- LOCAL_CONNECTION_FAILED — La connexion locale à Amazon a échoué. GameLift
- NETWORK_NOT_INITIALIZED — Le réseau n'a pas été initialisé.
- GAMESESSION_ID_NOT_SET — L'ID de session de jeu n'a pas été défini.
- MAUVAISE DEMANDE D'EXCEPTION
- EXCEPTION DE SERVICE INTERNE
- ALREADY_INITIALIZED — Le GameLift serveur ou le client Amazon a déjà été initialisé avec Initialize ().
- FLEET_MISMATCH — La flotte cible ne correspond pas à la flotte d'une GameSession ou d'une PlayerSession.
- GAMELIFT_CLIENT_NOT_INITIALIZED — Le client Amazon n'a pas été initialisé. GameLift
- GAMELIFT_SERVER_NOT_INITIALIZED — Le serveur Amazon n'a pas été initialisé. GameLift
- GAME_SESSION_ENDED_FAILED — Le SDK GameLift Amazon Server n'a pas pu contacter le service pour signaler la fin de la session de jeu.
- GAME_SESSION_NOT_READY — La session de jeu GameLift Amazon Server n'a pas été activée.
- GAME_SESSION_READY_FAILED — Le SDK GameLift Amazon Server n'a pas pu contacter le service pour signaler que la session de jeu était prête.
- INITIALIZATION_MISMATCH — Une méthode client a été appelée après Server : :Initialize (), ou vice versa.
- NOT_INITIALIZED — Le GameLift serveur ou le client Amazon n'a pas été initialisé avec Initialize ().
- NO_TARGET_ALIASID_SET — Aucun AliasID cible n'a été défini.
- NO_TARGET_FLEET_SET — Aucune flotte cible n'a été définie.
- PROCESS_ENDING_FAILED — Le SDK GameLift Amazon Server n'a pas pu contacter le service pour signaler la fin du processus.

- PROCESS_NOT_ACTIVE — Le processus du serveur n'est pas encore actif, n'est pas lié à un et ne peut ni GameSession accepter ni traiter. PlayerSessions
- PROCESS_NOT_READY — Le processus du serveur n'est pas encore prêt à être activé.
- PROCESS_READY_FAILED — Le SDK Amazon GameLift Server n'a pas pu contacter le service pour signaler que le processus était prêt.
- SDK_VERSION_DETECTION_FAILED — La détection de la version du SDK a échoué.
- STX_CALL_FAILED — Un appel au composant principal du serveur XSTx a échoué.
- STX_INITIALIZATION_FAILED — Le composant principal du serveur XSTx n'a pas pu être initialisé.
- UNEXPECTED_PLAYER_SESSION — Le serveur a détecté une session de joueur non enregistrée.
- ÉCHEC DE CONNEXION WEBSOCKET
- WEBSOCKET_CONNECT_FAILURE_FORBIDDEN
- WEBSOCKET_CONNECT_FAILURE_INVALID_URL
- WEBSOCKET_CONNECT_FAILURE_TIMEOUT
- WEBSOCKET_RETRIABLE_SEND_MESSAGE_FAILURE — Échec récupérable lors de l'envoi d'un message au service. GameLift WebSocket
- WEBSOCKET_SEND_MESSAGE_FAILURE — Impossible d'envoyer un message au service. GameLift WebSocket
- MATCH_BACKFILL_REQUEST_VALIDATION — La validation de la demande a échoué.
- PLAYER_SESSION_REQUEST_VALIDATION — La validation de la demande a échoué.

E PlayerSessionCreationPolicy

Valeur de chaîne indiquant si la session de jeu accepte ou non de nouveaux joueurs. Les valeurs valides sont les suivantes :

- ACCEPT_ALL — Accepte toutes les sessions de nouveaux joueurs.
- DENY_ALL — Refuse toutes les sessions de nouveaux joueurs.
- NOT_SET — La session de jeu n'est pas configurée pour accepter ou refuser les sessions de nouveaux joueurs.

E PlayerSessionStatus

- ACTIF

- TERMINÉ
- NOT_SET
- RÉSERVÉ
- DÉLAI EXPIRÉ

Référence du SDK GameLift 3.x du serveur Amazon Unreal Engine

Vous pouvez utiliser cette référence du SDK pour serveur Amazon GameLift Unreal Engine 3.x pour vous aider à préparer votre jeu multijoueur en vue de son utilisation avec Amazon. GameLift Pour plus de détails sur le processus d'intégration, consultezAjoutez Amazon GameLift à votre serveur de jeu.

Rubriques

- Référence GameLift du SDK Amazon Server pour Unreal Engine : Actions
- Référence GameLift du SDK Amazon Server pour Unreal Engine : types de données

Référence GameLift du SDK Amazon Server pour Unreal Engine : Actions

Cette référence GameLift du SDK Amazon Server peut vous aider à préparer vos projets de jeux Unreal Engine à utiliser avec Amazon. GameLift Pour plus d'informations sur le processus d'intégration, consultez Ajoutez Amazon GameLift à votre serveur de jeu.

Cette API est définie dans `GameLiftServerSDK.h` et `GameLiftServerSDKModels.h`.

Pour configurer le plug-in Unreal Engine et consulter des exemples de code Intégrer Amazon GameLift dans un projet Unreal Engine.

- Actions
- Types de données

AcceptPlayerSession()

Informe le GameLift service Amazon qu'un joueur avec l'identifiant de session de joueur spécifié s'est connecté au processus du serveur et doit être validé. Amazon GameLift vérifie que l'identifiant de session du joueur est valide, c'est-à-dire que l'identifiant du joueur a réservé un emplacement de joueur dans la session de jeu. Une fois la validation effectuée, GameLift Amazon fait passer le statut de la machine à sous RÉSERVÉ à ACTIVE.

Syntaxe

```
FGameLiftGenericOutcome AcceptPlayerSession(const FString& playerSessionId)
```

Paramètres

playerSessionId

Identifiant unique émis par le GameLift service Amazon en réponse à un appel à l'action CreatePlayerSessionde l'GameLiftAPI Amazon du AWS SDK. Le client du jeu fait référence à cet identifiant lors de la connexion au processus du serveur.

Type : FString

Obligatoire : oui

Valeur renvoyée

Renvoie un résultat générique composé d'un succès ou d'un échec avec un message d'erreur.

ActivateGameSession()

Indique au GameLift service Amazon que le processus du serveur a activé une session de jeu et qu'il est désormais prêt à recevoir les connexions des joueurs. Cette action doit être appelée dans le cadre de la fonction de rappel `onStartGameSession()`, une fois que l'initialisation de toutes les sessions de jeu est terminée.

Syntaxe

```
FGameLiftGenericOutcome ActivateGameSession()
```

Paramètres

Cette action n'a aucun paramètre.

Valeur renvoyée

Renvoie un résultat générique composé d'un succès ou d'un échec avec un message d'erreur.

DescribePlayerSessions()

Récupère les données de session de joueur, y compris les paramètres, les métadonnées de session et les données de joueur. Utilisez cette action pour obtenir des informations pour une seule session

de joueur, pour toutes les sessions de joueur d'une session de jeu ou pour toutes les sessions de joueur associées à un seul ID de joueur.

Syntaxe

```
FGameLiftDescribePlayerSessionsOutcome DescribePlayerSessions(const
 FGameLiftDescribePlayerSessionsRequest &describePlayerSessionsRequest)
```

Paramètres

describePlayerSessionsDemande

Objet F DescribePlayerSessionsRequest décrivant les sessions de joueur à récupérer.

Obligatoire : oui

Valeur renvoyée

En cas de réussite, renvoie un objet F DescribePlayerSessionsRequest qui contient un ensemble d'objets de session de joueur correspondant aux paramètres de la demande. Les objets de session du joueur ont une structure identique au type de PlayerSessiondonnées de l'GameLiftAPI Amazon du AWS SDK.

GetGameSessionId()

Extrait l'ID de la session de jeu actuellement hébergée par le processus serveur, si ce dernier est actif.

Syntaxe

```
FGameLiftStringOutcome GetGameSessionId()
```

Paramètres

Cette action n'a aucun paramètre.

Valeur renvoyée

En cas de réussite, renvoie l'ID de session de jeu en tant qu'objet `FGameLiftStringOutcome`. En cas d'échec, renvoie un message d'erreur.

GetInstanceCertificate()

Récupère l'emplacement du fichier d'un certificat TLS codé PEM associé au parc et à ses instances. AWS Certificate Managergénère ce certificat lorsque vous créez une nouvelle flotte avec la configuration du certificat définie sur GENERATED. Utilisez ce certificat pour établir une connexion sécurisée avec un client de jeu et pour chiffrer la communication client/serveur.

Syntaxe

```
FGameLiftGetInstanceCertificateOutcome GetInstanceCertificate()
```

Paramètres

Cette action n'a aucun paramètre.

Valeur renvoyée

En cas de succès, renvoie un `GetInstanceCertificateOutcome` objet contenant l'emplacement du fichier de certificat TLS et de la chaîne de certificats du parc, qui sont stockés sur l'instance. Un fichier de certificat racine, extrait de la chaîne de certificats, est également stocké sur l'instance. En cas d'échec, renvoie un message d'erreur.

Pour plus d'informations sur le certificat et les données de la chaîne de certificats, consultez la section Éléments de GetCertificate réponse dans la référence de l'AWS Certificate ManagerAPI.

GetSdkVersion()

Renvoie le numéro de version actuel du kit SDK intégré dans le processus serveur.

Syntaxe

```
FGameLiftStringOutcome GetSdkVersion();
```

Paramètres

Cette action n'a aucun paramètre.

Valeur renvoyée

En cas de réussite, renvoie la version actuelle du kit SDK en tant qu'objet `FGameLiftStringOutcome`. La chaîne renvoyée inclut uniquement le numéro de version (par exemple « 3.1.5 »). En cas d'échec, renvoie un message d'erreur.

Exemple

```
Aws::GameLift::AwsStringOutcome SdkVersionOutcome =
    Aws::GameLift::Server::GetSdkVersion();
```

InitSDK()

Initialise le GameLift SDK Amazon. Cette méthode doit être appelée au lancement, avant toute autre initialisation GameLift liée à Amazon.

Syntaxe

```
FGameLiftGenericOutcome InitSDK()
```

Paramètres

Cette action n'a aucun paramètre.

Valeur renvoyée

Renvoie un résultat générique composé d'un succès ou d'un échec avec un message d'erreur.

ProcessEnding()

Informe le GameLift service Amazon que le processus du serveur est en train de s'arrêter. Cette méthode doit être appelée après toutes les autres tâches de nettoyage, y compris l'arrêt de toutes les sessions de jeu actives. Cette méthode doit quitter avec un code de sortie 0 ; un code de sortie différent de 0 génère un message d'événement indiquant que le processus ne s'est pas fermé correctement.

Syntaxe

```
FGameLiftGenericOutcome ProcessEnding()
```

Paramètres

Cette action n'a aucun paramètre.

Valeur renvoyée

Renvoie un résultat générique composé d'un succès ou d'un échec avec un message d'erreur.

ProcessReady()

Indique au GameLift service Amazon que le processus du serveur est prêt à héberger des sessions de jeu. Appelez cette méthode après avoir invoqué InitSDK() et terminé avec succès les tâches de configuration requises pour que le processus serveur puisse héberger une session de jeu. Cette méthode ne doit être appelée qu'une seule fois par processus.

Syntaxe

```
FGameLiftGenericOutcome ProcessReady(FProcessParameters &processParameters)
```

Paramètres

F ProcessParameters

Objet F ProcessParameters communiquant les informations suivantes relatives au processus serveur :

- Noms des méthodes de rappel, implémentées dans le code du serveur de jeu, que le GameLift service Amazon invoque pour communiquer avec le processus du serveur.
- Numéro de port sur lequel le processus serveur écoute.
- Chemin d'accès à tous les fichiers spécifiques à une session de jeu que vous souhaitez qu'Amazon capture GameLift et stocke.

Obligatoire : oui

Valeur renvoyée

Renvoie un résultat générique composé d'un succès ou d'un échec avec un message d'erreur.

Exemple

Consultez l'exemple de code dans Utilisation du plug-in Unreal Engine.

RemovePlayerSession()

Informe le GameLift service Amazon qu'un joueur avec l'identifiant de session de joueur spécifié s'est déconnecté du processus serveur. En réponse, Amazon GameLift change le créneau de joueur en disponible, ce qui permet de l'attribuer à un nouveau joueur.

Syntaxe

```
FGameLiftGenericOutcome RemovePlayerSession(const FString& playerSessionId)
```

Paramètres

playerSessionId

Identifiant unique émis par le GameLift service Amazon en réponse à un appel à l'action CreatePlayerSessionde l'GameLiftAPI Amazon du AWS SDK. Le client du jeu fait référence à cet identifiant lors de la connexion au processus du serveur.

Type : FString

Obligatoire : oui

Valeur renvoyée

Renvoie un résultat générique composé d'un succès ou d'un échec avec un message d'erreur.

StartMatchBackfill()

Envoie une demande de recherche de nouveaux joueurs pour des emplacements ouverts dans une session de jeu créée avec FlexMatch. Voir également l'action du AWS SDK StartMatchBackfill(). Avec cette action, les requêtes de renvoi de correspondance peuvent être initiées par processus de serveur de jeu qui héberge la session de jeu. En savoir plus sur la fonction de FlexMatch remblayage.

Cette action est asynchrone. Si de nouveaux joueurs sont jumelés avec succès, le GameLift service Amazon fournit des données de matchmaking mises à jour à l'aide de la fonction de rappel. `OnUpdateGameSession()`

Un processus de serveur ne peut comporter qu'une seule requête de renvoi de correspondance à la fois. Pour envoyer une nouvelle requête, appelez d'abord StopMatchBackfill() pour annuler la requête d'origine.

Syntaxe

```
FGameLiftStringOutcome StartMatchBackfill (FStartMatchBackfillRequest
 &startBackfillRequest);
```

Paramètres

F StartMatchBackfillRequest

Objet F StartMatchBackfillRequest qui communique les informations suivantes :

- ID de ticket à attribuer à la requête de renvoi. Ces informations sont facultatives ; si aucun identifiant n'est fourni, Amazon en GameLift générera un automatiquement.
- Matchmaker auquel envoyer la requête. L'ARN de configuration complet est obligatoire. Cette valeur peut être acquise à partir des données matchmaker de la session de jeu.
- ID de la session de jeu en cours de renvoi.
- Données de correspondance disponibles pour les joueurs actuels de la session de jeu.

Obligatoire : oui

Valeur renvoyée

En cas de réussite, renvoie le ticket de renvoi de correspondance sous la forme d'un objet `FGameLiftStringOutcome`. En cas d'échec, renvoie un message d'erreur. L'état du ticket peut être suivi à l'aide de l'action du AWS SDK DescribeMatchmaking().

StopMatchBackfill()

Annule une requête de renvoi de correspondance active qui a été créée avec StartMatchBackfill(). Voir également l'action du AWS SDK StopMatchmaking(). En savoir plus sur la fonction de FlexMatch remblayage.

Syntaxe

```
FGameLiftGenericOutcome StopMatchBackfill (FStopMatchBackfillRequest
 &stopBackfillRequest);
```

Paramètres

StopMatchBackfillRequest

Objet F StopMatchBackfillRequest qui identifie le ticket de correspondance à annuler :

- Identifiant de ticket attribué à la requête de renvoi en cours d'annulation
- matchmaker auquel a été envoyée la requête

- session de jeu associée à la requête de renvoi

Obligatoire : oui

Valeur renvoyée

Renvoie un résultat générique composé d'un succès ou d'un échec avec un message d'erreur.

TerminateGameSession()

Cette méthode est obsolète depuis la version 4.0.1. Au lieu de cela, le processus du serveur doit appeler ProcessEnding() après la fin d'une session de jeu.

Informe le GameLift service Amazon que le processus du serveur a mis fin à la session de jeu en cours. Cette action est appelée lorsque le processus du serveur reste actif et prêt à héberger une nouvelle session de jeu. Il ne doit être appelé qu'une fois la procédure de fin de session de jeu terminée, car il indique à Amazon GameLift que le processus du serveur est immédiatement disponible pour héberger une nouvelle session de jeu.

Cette action n'est pas appelée si le processus du serveur doit être arrêté après la fin de la session de jeu. Appelez plutôt ProcessEnding() pour signaler que la session de jeu et le processus du serveur se terminent.

Syntaxe

```
FGameLiftGenericOutcome TerminateGameSession()
```

Paramètres

Cette action n'a aucun paramètre.

Valeur renvoyée

Renvoie un résultat générique composé d'un succès ou d'un échec avec un message d'erreur.

UpdatePlayerSessionCreationPolicy()

Met à jour la capacité de la session de jeu à accepter de nouvelles sessions de joueur. Une session de jeu peut être définie pour accepter ou refuser toutes les nouvelles sessions joueur. (Voir également l'`UpdateGameSession()`action dans la référence de l'API Amazon GameLift Service).

Syntaxe

```
FGameLiftGenericOutcome UpdatePlayerSessionCreationPolicy(EPlayerSessionCreationPolicy
 policy)
```

Paramètres

Politique

Valeur indiquant si la session de jeu accepte ou non de nouveaux joueurs.

Type : enum `EPlayerSessionCreationPolicy`. Les valeurs valides sont les suivantes :

- ACCEPT_ALL — Accepte toutes les sessions des nouveaux joueurs.
- DENY_ALL — Refuser toutes les sessions des nouveaux joueurs.

Obligatoire : oui

Valeur renvoyée

Renvoie un résultat générique composé d'un succès ou d'un échec avec un message d'erreur.

Référence GameLift du SDK Amazon Server pour Unreal Engine : types de données

Cette référence GameLift du SDK Amazon Server peut vous aider à préparer vos projets de jeux Unreal Engine à utiliser avec Amazon. GameLift Pour plus d'informations sur le processus d'intégration, consultez Ajoutez Amazon GameLift à votre serveur de jeu.

Cette API est définie dans `GameLiftServerSDK.h` et `GameLiftServerSDKModels.h`.

Pour configurer le plug-in Unreal Engine et consulter des exemples de code Intégrer Amazon GameLift dans un projet Unreal Engine.

- Actions
- Types de données

F DescribePlayerSessionsRequest

Ce type de données est utilisé pour spécifier les sessions de joueur à récupérer. Vous pouvez l'utiliser comme suit :

- Fournissez un PlayerSessionId pour demander une session de joueur spécifique.
- Indiquez un GameSessionId pour demander toutes les sessions des joueurs pendant la session de jeu spécifiée.
- Indiquez un PlayerId pour demander toutes les sessions du joueur spécifié.

Pour les volumes importants de sessions de joueur, utilisez les paramètres de pagination pour récupérer les résultats en tant que blocs séquentiels.

Table des matières

GameSessionId

Identifiant de session de jeu unique. Utilisez ce paramètre pour demander toutes les sessions de joueur pour la session de jeu spécifiée. Le format de l'ID de session de jeu est le suivant : `arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string>`. La valeur de **<**ID string**>** peut être une chaîne d'ID personnalisée ou (si spécifiée lors de la création de la session de jeu) une chaîne générée automatiquement.

Type : chaîne

Obligatoire : non

Limite

Nombre maximum de résultats à renvoyer. Utilisez ce paramètre avec NextTokenpour obtenir des résultats sous la forme d'un ensemble de pages séquentielles. Si un ID de session de joueur est spécifié, ce paramètre est ignoré.

Type : entier

Obligatoire : non

NextToken

Jeton indiquant le début de la prochaine page séquentielle de résultats. Utilisez le jeton qui est renvoyé par un appel précédent à cette action. Pour spécifier le début de l'ensemble de résultats, ne spécifiez aucune valeur. Si un ID de session de joueur est spécifié, ce paramètre est ignoré.

Type : chaîne

Obligatoire : non

PlayerId

Identifiant unique pour un joueur. Les ID de joueur sont définis par le développeur. Consultez Générer des identifiants de joueurs.

Type : chaîne

Obligatoire : non

PlayerSessionId

Identifiant unique d'une session de joueur.

Type : chaîne

Obligatoire : non

PlayerSessionStatusFilter

État de session de joueur pour filtrer les résultats. Les états de session de joueur possibles sont les suivants :

- RESERVED - La demande de session de joueur a été reçue, mais le joueur ne s'est pas encore connecté au processus serveur et/ou n'a pas encore été validé.
- ACTIVE - Le joueur a été validé par le processus serveur et est actuellement connecté.
- COMPLETED - La connexion du joueur a été abandonnée.
- TIMEDOUT - Une demande de session de joueur a été reçue, mais le joueur ne s'est pas connecté et/ou n'a pas été validé avant l'expiration du délai (60 secondes).

Type : chaîne

Obligatoire : non

F ProcessParameters

Ce type de données contient l'ensemble des paramètres envoyés au GameLift service Amazon lors d'un ProcessReady() appel.

Table des matières

port

Numéro de port sur lequel le processus serveur écoute les nouvelles connexions de joueur. La valeur doit être comprise dans la plage de ports configurée pour toutes les flottes déployant cette

version de génération du serveur de jeux. Ce numéro de port est inclus dans les objets de session de jeu et de session de joueur, que les sessions de jeu utilisent pour se connecter à un processus serveur.

Type : entier

Obligatoire : oui

logParameters

Objet comportant une liste de chemins de répertoires vers les fichiers journaux de sessions de jeu.

Type : TArray<FString>

Obligatoire : non

onStartGameSéance

Nom de la fonction de rappel invoquée par le GameLift service Amazon pour activer une nouvelle session de jeu. Amazon GameLift appelle cette fonction en réponse à la demande du client CreateGameSession. La fonction de rappel prend un GameSessionobjet (défini dans la référence de l'API Amazon GameLift Service).

Type : F OnStartGameSession

Obligatoire : oui

onProcessTerminate

Nom de la fonction de rappel invoquée par le GameLift service Amazon pour forcer l'arrêt du processus serveur. Après avoir appelé cette fonction, Amazon GameLift attend cinq minutes que le processus serveur s'arrête et répond par un ProcessEnding() appel avant d'arrêter le processus serveur.

Type : F SimpleDelegate

Obligatoire : non

onHealthCheck

Nom de la fonction de rappel invoquée par le GameLift service Amazon pour demander un rapport d'état de santé au processus serveur. Amazon GameLift appelle cette fonction toutes les 60 secondes. Après avoir appelé cette fonction, Amazon GameLift attend une réponse pendant 60

secondes et, si aucune réponse n'est reçue, enregistre le processus du serveur comme étant défectueux.

Type : F OnHealthCheck

Obligatoire : non

onUpdateGameSéance

Nom de la fonction de rappel invoquée par le GameLift service Amazon pour transmettre un objet de session de jeu mis à jour au processus du serveur. Amazon GameLift fait appel à cette fonction lorsqu'une demande de remplissage de correspondance a été traitée afin de fournir des données de matchmaking mises à jour. Il transmet un GameSessionobjet, une mise à jour de statut (`updateReason`) et l'identifiant du ticket de remplissage du match.

Type : F OnUpdateGameSession

Obligatoire : non

F StartMatchBackfillRequest

Ce type de données est utilisé pour envoyer une requête de renvoi de correspondance. Les informations sont communiquées au GameLift service Amazon lors d'un StartMatchBackfill() appel.

Table des matières

GameSessionArn

Identifiant de session de jeu unique. L'action d'API GetGameSessionId() renvoie l'identifiant au format ARN.

Type : FString

Obligatoire : oui

MatchmakingConfigurationArn

Identifiant unique, sous la forme d'un ARN, pour le matchmaker à utiliser pour cette requête. Pour trouver le matchmaker qui a été utilisé pour créer la session de jeu d'origine, recherchez dans l'objet de session de jeu, dans la propriété de données de matchmaker. Pour en savoir plus sur les données de matchmaking, consultez la section Travailler avec les données de matchmaking.

Type : FString

Obligatoire : oui

Joueurs

Ensemble de données représentant tous les joueurs qui sont actuellement dans la session de jeu. Le matchmaker utilise ces informations pour rechercher de nouveaux joueurs qui constituent de bonnes correspondances pour les joueurs actuels. Consultez le guide de référence des GameLift API Amazon pour obtenir une description du format d'objet Player. Pour trouver des attributs de joueur, des identifiants et des affectations d'équipe, recherchez dans l'objet de session de jeu, dans la propriété des données de matchmaker. Si une latence est utilisée par le matchmaker, collectez la latence mise à jour pour la région actuelle et incluez-la dans les données de chaque joueur.

Type : TArray<FPlayer>

Obligatoire : oui

TicketId

Identifiant unique pour une correspondance ou un ticket de requête de renvoi de correspondance. Si aucune valeur n'est fournie ici, Amazon en GameLift générera une sous la forme d'un UUID. Utilisez cet identifiant pour suivre l'état du ticket de renvoi de correspondance ou annuler la requête si nécessaire.

Type : FString

Obligatoire : non

F StopMatchBackfillRequest

Ce type de données est utilisé pour annuler une demande de renvoi de correspondance. Les informations sont communiquées au GameLift service Amazon lors d'un StopMatchBackfill() appel.

Table des matières

GameSessionArn

Identifiant de session de jeu unique associé à la requête en cours d'annulation.

Type : FString

Obligatoire : oui

MatchmakingConfigurationArn

Identifiant unique du matchmaker auquel cette requête a été envoyée.

Type : FString

Obligatoire : oui

TicketId

Identifiant unique d'un ticket de requête de correspondance à annuler.

Type : FString

Obligatoire : oui

Événements de placement de sessions de jeu

Amazon GameLift émet des événements pour chaque demande de placement de session de jeu au fur et à mesure de son traitement. Vous pouvez publier ces événements sur une rubrique Amazon SNS, comme décrit dans. Configurer une notification d'événement pour le placement des sessions de jeu Ces événements sont également transmis à Amazon CloudWatch Events en temps quasi réel et dans la mesure du possible.

Cette rubrique décrit la structure des événements de placement de session de jeu et fournit un exemple pour chaque type d'événement. Pour plus d'informations sur le statut des demandes de placement de session de jeu, consultez GameSessionPlacementle Amazon GameLift API Reference.

Syntaxe des événements de placement

Les événements sont représentés sous la forme d'objets JSON. La structure des événements est conforme au modèle des CloudWatch événements, avec des champs de haut niveau similaires et des détails spécifiques au service.

Les champs de niveau supérieur sont les suivants (voir le modèle d'événement pour plus de détails) :

version

Ce champ est toujours défini sur 0 (zéro).

id

Identifiant de suivi unique pour l'événement.

detail-type

La valeur est toujours`GameLift Queue Placement Event`.

source

La valeur est toujours`aws.gamelift`.

compte

Le AWS compte utilisé pour gérer Amazon GameLift.

time

Horodatage de l'événement.

region

AWS Région dans laquelle la demande de placement est traitée. Il s'agit de la région dans laquelle se trouve la file d'attente des sessions de jeu en cours d'utilisation.

resources

Valeur ARN de la file d'attente de session de jeu qui traite la demande de placement.

PlacementFulfilled

La demande de placement a été traitée avec succès. Une nouvelle session de jeu a été démarrée et de nouvelles sessions de joueur ont été créées pour chaque joueur répertorié dans la demande de placement de session de jeu. Les informations de connexion du joueur sont disponibles.

Syntaxe détaillée :

Identifiant du lieu

Identifiant unique attribué à la demande de placement de session de jeu.

port

Le numéro de port de la nouvelle session de jeu.

gameSessionArn

L'identifiant ARN de la nouvelle session de jeu.

ipAddress

Adresse IP de la session de jeu.

Nom DNS

Identifiant DNS attribué à l'instance qui exécute la nouvelle session de jeu. Le format de valeur est différent selon que l'instance qui exécute la session de jeu est compatible TLS ou non. Lorsqu'ils se connectent à une session de jeu sur une flotte compatible TLS, les joueurs doivent utiliser le nom DNS et non l'adresse IP.

Flottes compatibles TLS :. `<unique identifier>.<region identifier>.amazongamelift.com`

Flottes non compatibles TLS :. `ec2-<unique identifier>.compute.amazonaws.com`

startTime

Horodatage indiquant à quel moment cette demande a été placée dans la file d'attente.

endTime

Horodatage indiquant la date à laquelle cette demande a été satisfaite.

gameSessionRegion

AWS Région de la flotte qui héberge la session de jeu. Cela correspond au jeton de région dans le`GameSessionArn`.

placedPlayerSessions

L'ensemble des sessions de joueur créées pour chaque joueur dans la demande de placement de session de jeu.

Exemple

```
{
  "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

La demande de placement a été annulée par un appel au GameLift service StopGameSessionPlacement.

Détail :

Identifiant du lieu

Identifiant unique attribué à la demande de placement de session de jeu.

startTime

Horodatage indiquant à quel moment cette demande a été placée dans la file d'attente.

endTime

Horodatage indiquant la date à laquelle cette demande a été annulée.

Exemple

```
{
  "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

Le placement de la session de jeu n'a pas été effectué correctement avant l'expiration du délai de la file d'attente. La demande de placement peut être soumise à nouveau si nécessaire.

Détail :

Identifiant du lieu

Identifiant unique attribué à la demande de placement de session de jeu.

startTime

Horodatage indiquant à quel moment cette demande a été placée dans la file d'attente.

endTime

Horodatage indiquant la date à laquelle cette demande a été annulée.

Exemple

```
{
  "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 n' GameLift a pas été en mesure de répondre à la demande de session de jeu. Cela est généralement dû à une erreur interne inattendue. La demande de placement peut être soumise à nouveau si nécessaire.

Détail :

Identifiant du lieu

Identifiant unique attribué à la demande de placement de session de jeu.

startTime

Horodatage indiquant à quel moment cette demande a été placée dans la file d'attente.

endTime

Horodatage indiquant à quel moment cette demande a échoué.

Exemple

```
{
  "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"
  }
}
```

Génération d'estimations GameLift de prix Amazon

AvecAWS Pricing Calculator, vous pouvez créer une estimation de prix pour Amazon GameLift. Vous n'avez pas besoin de connaissances approfondies AWS pour utiliser le calculateur. Compte AWS

AWS Pricing CalculatorLe calculateur vous guide à travers les décisions qui influent sur les coûts de service afin de vous donner une idée du coût que GameLift pourrait vous coûter Amazon pour votre projet de jeu. Si vous ne savez pas encore comment vous comptez utiliser AmazonGameLift, utilisez les valeurs par défaut pour générer une estimation. Lors de la planification de l'utilisation en production, le calculateur peut vous aider à tester des scénarios potentiels et à générer des estimations plus précises.

Vous pouvez l'utiliser AWS Pricing Calculator pour générer des estimations pour les options GameLift d'hébergement Amazon suivantes :

- Estimation de l'GameLifthébergement Amazon
- Estimation de la version GameLift autonome d'Amazon FlexMatch

Estimation de l'GameLifthébergement Amazon

Cette option fournit une estimation du coût de l'hébergement de vos jeux sur les serveurs GameLift gérés par Amazon, y compris les coûts d'utilisation des instances de serveur et de transfert de données. FlexMatchle matchmaking est inclus dans le coût de l'hébergement GameLift géré par Amazon.

Si vous hébergez ou prévoyez d'héberger des serveurs de jeu dans plusieurs AWS régions ou sur plusieurs types d'instances, créez une estimation pour chaque région et type d'instance.

GameLiftInstances Amazon

Cette section vous permet d'estimer le type et le nombre de ressources informatiques dont vous avez besoin pour héberger des sessions de jeu pour vos joueurs. Amazon GameLift utilise les instances Amazon Elastic Compute Cloud (Amazon EC2) pour gérer les serveurs de jeux. Dans AmazonGameLift, vous déployez un parc d'instances avec un type d'instance et un système d'exploitation spécifiques. Si vous possédez ou prévoyez d'avoir plusieurs flottes, créez une estimation pour chaque flotte.

Pour commencer, ouvrez la GameLiftpage Configurer Amazon deAWS Pricing Calculator. Ajoutez une description, choisissez une région, puis choisissez Estimer l'GameLifthébergement Amazon (Instance + Data Transfer Out). Sous GameLiftInstances Amazon, renseignez les champs suivants :

- Nombre maximal de joueurs simultanés (pic de CCU)

  Il s'agit du nombre maximum de joueurs pouvant se connecter simultanément à vos serveurs de jeu. Ce champ indique la capacité d'hébergement dont Amazon GameLift a besoin pour répondre à la demande maximale des joueurs. Entrez le pic quotidien de joueurs que vous comptez héberger en utilisant les instances de AWS la région que vous avez choisie.

  Par exemple, si vous souhaitez autoriser 1 000 joueurs à se connecter à votre jeu à tout moment, conservez la valeur par défaut de**1000**.

- CCU moyen par heure en pourcentage du CCU quotidien de pointe

  Il s'agit du nombre moyen de joueurs simultanés par heure sur une période de 24 heures. Nous utilisons cette valeur pour estimer la capacité d'hébergement soutenue qu'Amazon GameLift doit maintenir pour vos joueurs. Si vous n'êtes pas sûr de la valeur en pourcentage à utiliser, conservez la valeur par défaut de **50** pourcentage. Pour les jeux dont la demande de joueurs est stable, nous vous recommandons de saisir une valeur de **70** pourcentage.

  Par exemple, si votre jeu affiche un CCU horaire moyen de 6 000 et un CCU maximal de 10 000, entrez la valeur du **60** pourcentage.

- Sessions de jeu par instance

  Il s'agit du nombre de sessions de jeu que chacune de vos instances de serveur de jeu peut héberger simultanément. Les facteurs qui peuvent influencer ce chiffre incluent les besoins en ressources de votre serveur de jeu, le nombre de joueurs à héberger lors de chaque session de jeu et les attentes des joueurs en matière de performances. Si vous connaissez le nombre de sessions de jeu simultanées pour votre jeu, saisissez cette valeur. Vous pouvez également conserver la valeur par défaut de**20**.

- Nombre de joueurs par session de jeu

  Il s'agit du nombre moyen de joueurs qui se connectent à une session de jeu, tel que défini dans la conception de votre jeu. Si vous avez des modes de jeu avec un nombre de joueurs différent, estimez le nombre moyen de joueurs par session de jeu sur l'ensemble de votre jeu. La valeur par défaut est **8**.

- % de mémoire tampon inactive de l'instance

Il s'agit du pourcentage de capacité d'hébergement inutilisée à conserver en réserve pour faire face aux pics soudains de demande des joueurs. La taille de la mémoire tampon est un pourcentage du nombre total d'instances d'un parc. La valeur par défaut est le **10** pourcentage.

Par exemple, avec une mémoire tampon inactive de 20 %, une flotte accueillant des joueurs disposant de 100 instances actives conserve 20 instances inactives.

- % d'instances Spot

  GameLiftLes flottes Amazon peuvent utiliser une combinaison d'instances à la demande et d'instances ponctuelles. Alors que les instances à la demande offrent une disponibilité plus fiable, les instances Spot constituent une alternative très rentable. Nous vous recommandons d'utiliser une combinaison pour optimiser à la fois les économies de coûts et la disponibilité. Pour plus d'informations sur la manière dont Amazon GameLift utilise les instances Spot, consultezInstances à la demande et instances ponctuelles.

  Dans ce champ, entrez le pourcentage d'instances Spot à maintenir dans un parc. Nous recommandons un pourcentage d'instances Spot compris entre 50 et 85 %. La valeur par défaut est le **50** pourcentage.

  Par exemple, si vous déployez une flotte de 100 instances et que vous spécifiez un **40** pourcentage, Amazon s'GameLiftefforce de maintenir 60 instances à la demande et 40 instances Spot.

- Type d'instance

  GameLiftLes flottes Amazon peuvent utiliser différents types d'instances Amazon EC2 dont la puissance de calcul, la mémoire, le stockage et les capacités réseau varient. Lorsque vous configurez une GameLift flotte Amazon, choisissez le type d'instance qui correspond le mieux aux besoins de votre jeu. Pour plus d'informations sur la sélection d'un type d'instance avec AmazonGameLift, consultezChoisir les ressources GameLift informatiques d'Amazon.

  Si vous connaissez le type d'instance que vous utilisez ou que vous prévoyez d'utiliser dans votre GameLift flotte Amazon, choisissez ce type. Si vous n'êtes pas sûr du type à choisir, pensez à choisir c5.large. Il s'agit d'un type à haute disponibilité avec une taille et des capacités moyennes.

- Système d'exploitation

  Ce champ indique le système d'exploitation sur lequel s'exécutent vos serveurs de jeu, Linux ou Windows. La valeur par défaut est Linux.

Transfert de données sortant (DTO)

Cette section vous permet d'estimer le coût du trafic entre vos clients de jeu et les serveurs de jeu. Les frais de transfert de données s'appliquent uniquement au trafic sortant. Le transfert de données entrantes est gratuit.

Sur la GameLiftpage Configurer Amazon deAWS Pricing Calculator, développez Transfert de données sortantes (DTO), puis renseignez les champs suivants :

- Type d'estimation DTO

  Vous pouvez choisir d'estimer le DTO de l'une des deux manières suivantes, en fonction de la façon dont vous suivez les transferts de données pour votre jeu.

  - Par mois (en Go) : si vous suivez le trafic mensuel de vos serveurs de jeu, choisissez ce type.
  - Par joueur — Si vous suivez le transfert de données par joueur, choisissez ce type. Il s'agit du type par défaut.

    Dans le champ suivant, vous estimez le DTO par joueur en fonction du nombre d'heures de jeu que vous avez calculé dans la section précédente.
- DTO par mois (en Go)

  Si vous avez choisi le type d'estimation DTO par mois (en Go), entrez votre consommation mensuelle estimée de DTO en Go pour chaque instance et par région.
- DTO par joueur

  Si vous avez choisi le type d'estimation du DTO par joueur, saisissez l'estimation de la consommation de DTO par joueur de votre jeu en Ko/s. La valeur par défaut est **4**.

Lorsque vous avez terminé de configurer votre estimation de GameLift prix Amazon, choisissez Ajouter à mon estimation. Pour plus d'informations sur la création et la gestion des estimations dansAWS Pricing Calculator, voir Créer une estimation, configurer un service et ajouter d'autres services dans le Guide de AWS Pricing Calculator l'utilisateur.

Estimation de la version GameLift autonome d'Amazon FlexMatch

Cette option fournit une estimation des coûts liés à l'utilisation du FlexMatch matchmaking en tant que service autonome tout en hébergeant vos jeux avec une autre solution de serveur de jeu. Cela inclut l'hébergement GameLift autogéré Amazon avec FleetIQ et l'hébergement sur sitepeer-to-peer,

ou les types de données primitifs de calcul dans le cloud. Les FlexMatch coûts autonomes sont basés sur la puissance de calcul utilisée.

Si vous avez ou prévoyez d'avoir plus d'un système de matchmaking dans différentes AWS régions, créez une estimation pour chaque région.

> **Note**
>
> Amazon GameLift FlexMatch est disponible dans les régions suivantes : USA Est (Virginie du Nord), USA Ouest (Oregon), Asie-Pacifique (Séoul), Asie-Pacifique (Sydney), Asie-Pacifique (Tokyo), Europe (Francfort), Europe (Irlande).

Pour commencer, ouvrez la GameLiftpage Configurer Amazon deAWS Pricing Calculator. Ajoutez une description, choisissez une région, puis choisissez Estimate Amazon GameLift Standalone. FlexMatch Sous Amazon GameLift FlexMatch, renseignez les champs suivants :

- Nombre maximal de joueurs simultanés (pic de CCU)

  Il s'agit du nombre maximum de joueurs pouvant se connecter simultanément à vos serveurs de jeu et demander le matchmaking. Entrez le pic quotidien de joueurs que vous comptez voir participer à des sessions de jeu dans la région que vous avez choisie.

  Par exemple, si vous souhaitez faire correspondre jusqu'à 1 000 joueurs à la fois, conservez la valeur par défaut de**`1000`**.
- CCU moyen par heure en pourcentage du CCU quotidien de pointe

  Il s'agit du nombre moyen de joueurs simultanés par heure sur une période de 24 heures. Cette valeur permet d'estimer le volume de vos demandes de matchmaking. Si vous n'êtes pas sûr de la valeur en pourcentage à utiliser, conservez la valeur par défaut de **`50`** pourcentage. Pour les jeux dont la demande de joueurs est stable, nous vous recommandons de saisir une valeur de **`70`** pourcentage.

  Par exemple, si votre jeu affiche un CCU horaire moyen de 6 000 et un CCU maximal de 10 000, entrez la valeur du **`60`** pourcentage.
- Nombre de joueurs par match

  Il s'agit du nombre moyen de joueurs participant à une session de jeu, tel que défini dans la conception de votre jeu. Si vous avez des modes de jeu avec un nombre de joueurs différent,

estimez le nombre moyen de joueurs par session de jeu sur l'ensemble de votre jeu. La valeur par défaut est **8**.

- Durée du jeu (en minutes)

  Il s'agit de la durée moyenne pendant laquelle les joueurs restent dans une session de jeu du début à la fin. Cette valeur permet de déterminer la fréquence à laquelle les joueurs peuvent avoir besoin d'un nouveau match. Entrez la durée moyenne de jeu en minutes pour vos joueurs. La valeur par défaut est **1**.

- Complexité des règles de matchmaking

  La complexité des règles de matchmaking fait référence au nombre et au type de règles que vous utilisez pour faire correspondre les joueurs. Le niveau de complexité de votre ensemble de règles permet de déterminer la puissance de calcul requise pour chaque match.

  - Complexité réduite — Choisissez cette option si votre ensemble de règles de matchmaking comprend peu de règles, utilise des types de règles plus simples (comme les règles de comparaison) et contient des règles qui permettent d'obtenir des correspondances réussies avec moins de tentatives.
  - Complexité accrue — Choisissez cette option si votre ensemble de règles de matchmaking comprend plusieurs règles, utilise des types de règles plus complexes (tels que des règles de distance ou de latence) et comporte des règles restrictives qui entraînent davantage d'échecs et nécessitent davantage de tentatives de mise en correspondance.

  Pour plus d'informations sur la complexité des règles et la tarification, consultez Amazon GameLift FlexMatch sur la page GameLift des tarifs Amazon.

Lorsque vous avez terminé de configurer votre estimation de GameLift FlexMatch prix Amazon, choisissez Ajouter à mon estimation. Pour plus d'informations sur la création et la gestion des estimations dansAWS Pricing Calculator, voir Créer une estimation, configurer un service et ajouter d'autres services dans le Guide de AWS Pricing Calculator l'utilisateur.

Quotas et régions prises en charge

Pour les quotas GameLift de service AWS Amazon, consultez la section GameLiftQuotas Amazon.

Pour plus d'informations sur les demandes d'augmentation des quotas pour les AWS ressources, consultez la section Quotas AWS de service.

Pour obtenir la liste des Amazon Régions AWS concernésGameLift, consultez la section GameLiftRégions Amazon.

Notes de GameLift publication d'Amazon

Les notes GameLift de publication d'Amazon fournissent des informations sur les nouvelles fonctionnalités, les mises à jour et les correctifs liés au service.

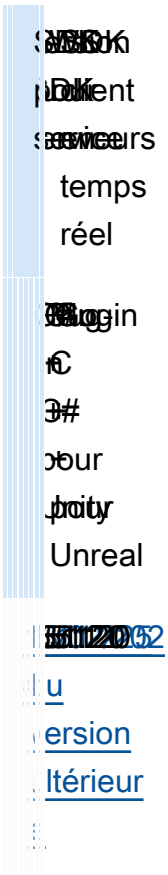
Versions SDK

Les tableaux suivants répertorient toutes les GameLift versions d'Amazon avec des informations sur la version du SDK. Il n'est pas nécessaire d'utiliser des SDK comparables pour les intégrations de votre serveur de jeu et de votre client. Toutefois, les versions antérieures d'un SDK peuvent ne pas prendre entièrement en charge les dernières fonctionnalités d'un autre SDK.

Pour plus d'informations sur GameLift les SDK Amazon, consultezAssistance au développement avec Amazon GameLift.

Pour obtenir les derniers GameLift SDK Amazon, consultez le site de téléchargement GameLiftdes SDK Amazon.

Version actuelle



Versions précédentes

Version du service	AWS SDK	SDK pour serveurs				SDK client en temps réel
		Plug-in C# pour Unity	C++	Plugin C++ pour Unreal	Go	
14/12/2023	1.11.225 ou version ultérieure	5.1.0	5.1.1	5.1.0	5.0.0	1.2.0
02/11/2023	1.11.193 ou version ultérieure	5.1.0	5.1.1	5.1.0	5.0.0	1.2.0
28/09/2023	1.11.144 ou version ultérieure	5.1.0	5.1.1	5.1.0	5.0.0	1.2.0
17/08/2023	1.11.144 ou version ultérieure	5.1.0	5.1.1	5.1.0	5.0.0	1.2.0
27/07/2023	1.11.111 ou version ultérieure	5.1.0	5.1.0	5.0.2	5.0.0	1.2.0
29/06/2023	1.11.111 ou version ultérieure		5.0.4	5.0.2	5.0.0	1.2.0

Version du service	AWS SDK	SDK pour serveurs				SDK client en temps réel
		Plug-in C# pour Unity	C++	Plugin C++ pour Unreal	Go	
15/06/2023	1.11.87 ou version ultérieure		5.0.4	5.0.2	5.0.0	1.2.0
25/05/2023	1.11.87 ou version ultérieure		5.0.3	5.0.2	5.0.0	1.2.0
20/04/2023	1.11.63 ou version ultérieure				5.0.0	1.2.0
13/04/2023	1.10.21 ou version ultérieure				5.0.0	1.2.0
09/02/2023	1.10.21 ou version ultérieure			3.4.0	5.0.0	1.2.0
31/01/2023	1.10.21 ou version ultérieure			3.4.0	5.0.0	1.2.0
01/12/2018	1.10.21 ou version ultérieure			3.4.0		1.2.0

Version du service	AWS SDK	SDK pour serveurs				SDK client en temps réel
		Plug-in C# pour Unity	C++	Plugin C++ pour Unreal	Go	
08-25	1.9.333 ou version ultérieure		3.4.2	3.4.0		1.2.0
2021-10-28	1.9.133 ou version ultérieure		3.4.2	3.4.0		1.2.0
03/06/2021	1.8.168 ou version ultérieure		3.4.2	3.4.0		1.2.0
23/03/2021	1.8.168 ou version ultérieure		3.4.1	3.3.3		1.1.0
16/03/2021	1.8.163 ou version ultérieure		3.4.1	3.3.3		1.1.0
09/02/2021	1.8.139 ou version ultérieure		3.4.1	3.3.3		1.1.0
22/12/2020	1.8.95 ou version ultérieure		3.4.1	3.3.3		1.1.0

Version du service	AWS SDK	SDK pour serveurs				SDK client en temps réel
		Plug-in C# pour Unity	C++	Plugin C++ pour Unreal	Go	
24/11/2020	1.8.95 ou version ultérieure		3.4.1	3.3.2		1.1.0
11 novembre 2020	1.8.36 ou version ultérieure		3.4.1	3.3.2		1.1.0
17/09/2020	1.8.36 ou version ultérieure		3.4.1	3.3.2		1.1.0
27/08/2020	1.7.310 ou version ultérieure		3.4.0	3.3.1		1.1.0
2020-04-16	1.7.310 ou version ultérieure		3.4.0	3.3.1		1.1.0
2020-04-02	1.7.310 ou version ultérieure		3.4.0			1.1.0
2019-12-19	1.7.249 ou version ultérieure		3.4.0			1.1.0

Version du service	AWS SDK	SDK pour serveurs				SDK client en temps réel
		Plug-in C# pour Unity	C++	Plugin C++ pour Unreal	Go	
2019-11-14	1.7.210 ou version ultérieure		3.4.0			1.1.0
2019-10-24	1.7.210 ou version ultérieure		3.4.0			1.1.0
2019-09-03	1.7.175 ou version ultérieure		3.4.0			1.1.0
2019-07-09	1.7.140 ou version ultérieure		3.3.0			1.0.0
25/04/2019	1.7.91 ou version ultérieure		3.3.0			1.0.0
2019-03-07	1.7.65 ou version ultérieure		3.3.0			
2019-02-07	1.7.45 ou version ultérieure		3.3.0			

Version du service	AWS SDK	SDK pour serveurs				SDK client en temps réel
		Plug-in C# pour Unity	C++	Plugin C++ pour Unreal	Go	
14/12/2018	1.6.20 ou version ultérieure		3.3.0			
2018-09-27	1.6.20 ou version ultérieure		3.2.1			
2018-06-14	1.4.47 ou version ultérieure		3.2.1			
2018-05-10	1.4.47 ou version ultérieure		3.2.1			
15/02/2018	1.3.58 ou version ultérieure		3.2.1			
08/02/2018	1.3.52 ou version ultérieure		3.2.0			
01/09/2017	1.1.43 ou version ultérieure		3.1.7			

Version du service	AWS SDK	SDK pour serveurs				SDK client en temps réel
		Plug-in C# pour Unity	C++	Plugin C++ pour Unreal	Go	
16/08/2017	1.1.31 ou version ultérieure		3.1.7			
2017-05-16	1.0.122 ou version ultérieure		3.1.5			
2017-04-11	1.0.103 ou version ultérieure		3.1.5			
21/02/2017	1.0.72 ou version ultérieure		3.1.5			
2016-11-18	1.0.31 ou version ultérieure		3.1.0			
2016-10-13	1.0.17 ou version ultérieure		3.1.0			
01/09/2016	0.14.9 ou version ultérieure		3.1.0			

Version du service	AWS SDK	SDK pour serveurs				SDK client en temps réel
		Plug-in C# pour Unity	C++	Plugin C++ pour Unreal	Go	
04/08/2016	0.12.16 ou version ultérieure		3,0.7			

Notes de mise à jour

Les notes de publication suivantes sont présentées par ordre chronologique, les dernières mises à jour étant répertoriées en premier. Amazon GameLift a été lancé pour la première fois en 2016. Pour les notes de mise à jour datées plus tôt que celles répertoriées ici, consultez les liens de date de publication dans Versions SDK.

24 avril 2024 : Amazon GameLift lance des flottes de conteneurs

Amazon propose GameLift désormais un aperçu des flottes de conteneurs, qui vous permettent d'améliorer la portabilité, l'évolutivité, la tolérance aux pannes et l'agilité.

Dans les flottes de conteneurs, les instances Amazon EC2 hébergent un ou plusieurs de vos conteneurs. Ces conteneurs incluent votre serveur de jeu ainsi que tout ce dont il a besoin, y compris les dépendances et les configurations. Les exemples de dépendances incluent les SDK et les progiciels. Une fois que vous avez chargé votre conteneur dans votre Amazon Elastic Container Registry privé, Amazon GameLift remplit votre flotte avec le conteneur.

Pour fonctionner dans une flotte de conteneurs, votre serveur de jeu doit fonctionner sous Linux et être intégré au SDK 5.x. Dans un parc de conteneurs, vous pouvez contrôler avec précision les ressources d'hébergement afin d'optimiser la consommation de ressources telles que les unités CPU et la mémoire. Vous pouvez également héberger plusieurs serveurs de jeu dans un conteneur afin de réduire l'utilisation des ressources.

Dans un parc de conteneurs, vous bénéficiez des mêmes avantages que les autres types de flottes, tels que les types d'instances à la demande, le dimensionnement (automatique et manuel), les files d'attente et le matchmaking. Vous obtenez également les mêmes indicateurs que les autres types de flottes, ainsi que de nouveaux indicateurs pour les conteneurs. Les flottes de conteneurs vous permettent de toucher les joueurs du monde entier dans les régions suivantes :

- ap-northeast-1
- ap-northeast-2
- ap-southeast-2
- eu-central-1
- eu-west-1
- us-east-1
- us-west-2

Pour atteindre encore plus de régions et de zones locales, créez des flottes de conteneurs multi-sites.

En savoir plus :

- Gestion de l'hébergement avec GameLift des conteneurs Amazon, Amazon GameLift Developer Guide
- CreateContainerGroupDefinition, Référence d' GameLift API Amazon

13 février 2024 : Amazon apporte GameLift des améliorations aux SDK et simplifie l'installation du GameLift plugin Amazon pour Unreal Engine

Versions du SDK mises à jour :

- Kit de développement logiciel Go Server, version 5.1.0
- SDK du serveur C#, version 5.1.2
- SDK pour serveur C++, version 5.1.2

Nous avons apporté les améliorations suivantes :

- Amélioration de la fiabilité du SDK en ajoutant une reconnexion automatique en cas d'interruption du réseau.

- [Go] Vous pouvez désormais appeler `InitSDK()` avec ou sans paramètres de serveur. Les serveurs de jeu qui s'exécutent sur des flottes EC2 GameLift gérées par Amazon lisent les paramètres du serveur directement à partir des variables d'environnement. Les serveurs de jeu des GameLift Anywhere flottes Amazon doivent appeler `InitSDK()` avec les paramètres du serveur.

Versions du plugin mises à jour :

- GameLift Plug-in Amazon pour Unreal Engine, version 1.1.0
- GameLift Plug-in Amazon pour Unity, version 2.1.0
- Plug-in SDK C++ Server pour Unreal, version 5.1.1
- Plug-in SDK du serveur C# pour Unity, version 5.1.2

Nous avons apporté les améliorations suivantes :

- [GameLift Plug-in Amazon pour Unreal Engine] Les instructions d'installation ont été mises à jour et l'emballage a été simplifié. Ce plugin inclut désormais la dernière version du SDK C++ Server pour Unreal.
- Les plug-ins ont été mis à niveau pour prendre en charge la dernière version du SDK GameLift du serveur.

En savoir plus :

- Intégration de jeux avec le GameLift plugin Amazon pour Unreal Engine, Amazon GameLift Developer Guide
- Téléchargements de GameLift plugins et de SDK Amazon

14 décembre 2023 : Amazon GameLift ajoute la possibilité de mettre à jour les propriétés de jeu des sessions de jeu actives

Vous avez déjà été en mesure de définir les propriétés du jeu lors de la création de sessions de jeu et de rechercher des propriétés spécifiques dans les sessions de jeu. Vous pouvez désormais également ajouter et mettre à jour ces propriétés dans une session de jeu active.

Par exemple, vos joueurs votent sur une carte sur laquelle ils veulent jouer. Votre client de jeu appelle `UpdateGameSession` pour modifier une `GameProperty` valeur en`{"Key": "map",`

`"Value":"jungle"}`. Votre jeu implémente ensuite la nouvelle carte pour les joueurs de la session de jeu.

Les administrateurs du jeu peuvent également récupérer des données utiles à partir des propriétés du jeu en utilisant `SearchGameSessions` cette opération. Par exemple, les administrateurs peuvent répertorier les sessions de jeu ayant une `Status` valeur de `ACTIVE` et cette propriété de jeu :`{"Key": "map", "Value":"desert"}`.

En savoir plus :

- the section called “Ajouter Amazon GameLift à un client de jeu”, Guide GameLift du développeur Amazon
- GameProperty, Référence d' GameLift API Amazon
- UpdateGameSession, Référence d' GameLift API Amazon
- SearchGameSessions, Référence d' GameLift API Amazon

21 novembre 2023 : Amazon GameLift lance le support pour les outils d'infrastructure en tant que code tels que Terraform et Pulumi optimisés par AWS Cloud Control API

Vous pouvez désormais gérer l'intégralité de votre pile de GameLift ressources Amazon à l'aide des outils d'infrastructure en tant que code (IaC). Ces outils incluent AWS CloudFormation, ainsi que des outils tiers tels que Terraform et Pulumi. Grâce à cette assistance supplémentaire, vous pouvez désormais vous concentrer sur le développement de votre jeu et tirer parti de DevOps stratégies pour prendre en charge la gestion des ressources, le CI/CD et le déploiement auprès de vos clients.

Vous pouvez également désormais approvisionner et configurer tous les types de GameLift ressources Amazon à l'aide de l'API AWS Cloud Control. Vous pouvez continuer à utiliser les ressources à l'aide GameLift des API Amazon ou des AWS CloudFormation modèles pour Amazon GameLift.

Pour plus de détails sur les GameLift ressources Amazon disponibles via IaC, consultez la référence du type de GameLift ressource Amazon. GameLift

En outre, vous pouvez désormais redimensionner automatiquement vos flottes à l'aide de AWS CloudFormation modèles ou de l'API AWS Cloud Control en utilisant la nouvelle propriété Fleet :`ScalingPolicies`.

L'API Cloud Control fournit aux développeurs un ensemble standard d'API pour créer, lire, mettre à jour, supprimer et répertorier des ressources (CRUDL) sur des centaines de AWS services et de nombreux outils tiers tels que Terraform et Pulumi.

En savoir plus :

- AWS CloudFormation
- AWS API de contrôle du cloud
- AWS Fournisseur CC Terraform
- Pulumi

16 novembre 2023 : Amazon GameLift met à jour le plugin autonome pour Unity

Versions du SDK mises à jour : GameLift plugin Amazon pour Unity, version 2.0.0

Le GameLift plugin Amazon pour Unity fournit des outils et des flux de travail qui simplifient les étapes nécessaires à la mise en service de votre jeu Unity pour l'hébergement dans le cloud avec Amazon GameLift. Amazon GameLift est un service entièrement géré qui permet aux développeurs de jeux de gérer et de faire évoluer des serveurs de jeux dédiés pour les jeux multijoueurs basés sur des sessions.

Avec cette version, le plugin pour Unity est mis à jour pour utiliser les dernières GameLift fonctionnalités d'Amazon, notamment la version 5.x du SDK pour le serveur et la prise en charge des tests locaux avec Amazon GameLift Anywhere. Le plugin est compatible avec les versions Unity 2021.3 LTS et 2022.3 LTS de Unity.

Les principales fonctionnalités du plugin incluent :

- Workflows d'interface utilisateur guidés dans l'éditeur Unity pour les scénarios suivants :
  - Testez l'intégration de votre jeu avec Amazon GameLift en utilisant votre station de travail locale comme hôte. Ce flux de travail vous permet de configurer une GameLift Anywhere flotte Amazon pour votre machine locale, de lancer des instances de votre serveur de jeu et de votre client, de demander une session de jeu via Amazon GameLift et de rejoindre le jeu.
  - Déployez une solution d'hébergement cloud pour votre serveur de jeu intégré avec Amazon EC2 GameLift géré par Amazon et des AWS ressources de support. Ce flux de travail vous aide à configurer votre jeu pour l'hébergement dans le cloud et propose trois scénarios de déploiement :
    - Déployez le serveur de jeu sur une seule flotte.

    - Déployez le serveur de jeu sur un ensemble de flottes de Spot à faible coût dans plusieurs AWS régions.
    - Déployez le serveur de jeu avec un FlexMatch système de matchmaking.
- Possibilité de configurer des profils utilisateur liés à un AWS compte utilisateur et de définir une AWS région par défaut. Vous pouvez gérer plusieurs profils pour travailler dans différents AWS comptes, utilisateurs de comptes et régions.
- Des fonctionnalités spéciales qui aident à rationaliser les processus GameLift d'intégration et de déploiement d'Amazon, notamment :
    - Chaque solution d'hébergement inclut des AWS ressources de support, notamment un pool d'utilisateurs Amazon Cognito qui fournit des identifiants de joueur uniques et une validation des joueurs. Les solutions incluent également un compartiment Amazon S3 pour le stockage, les notifications d'événements Amazon SNS, les AWS Lambda fonctions et d'autres ressources.
    - Pour le Anywhere flux de travail, le plugin automatise les paramètres de serveur requis.
    - Pour le flux de travail Amazon EC2, chaque solution de déploiement fournit un service client backend intégré utilisant les fonctions Lambda. Le service principal se situe entre le client du jeu et le GameLift service Amazon et gère tous les appels directs vers le GameLift service Amazon.
- Contenu destiné aux tests d'intégration, y compris les ressources et le code d'un exemple de jeu multijoueur simple illustrant l'intégration du serveur de jeu et du client de jeu.
- Documentation du plugin avec des conseils d'intégration détaillés et un exemple de code.

Tous les scénarios de déploiement, y compris pour Anywhere et pour les flottes Amazon EC2, utilisent des AWS CloudFormation modèles pour décrire et déployer les AWS ressources nécessaires à la solution de votre jeu. Ces modèles sont inclus dans le téléchargement du GameLift plugin Amazon. Vous pouvez les utiliser tels quels ou les personnaliser pour votre jeu.

En savoir plus :

- Guide du GameLift plugin Amazon pour Unity pour le SDK de serveur 5.x, Guide GameLift du développeur Amazon
- Téléchargez le plugin sur GitHub
- À propos de l' GameLift hébergement Amazon
- GameLift Forum Amazon

2 novembre 2023 : Amazon GameLift ajoute la prise en charge des informations d'identification partagées

Versions du SDK mises à jour : AWS SDK 1.11.193

La nouvelle fonctionnalité d'informations d'identification GameLift partagées d'Amazon permet aux applications déployées sur des flottes EC2 gérées d'interagir avec d'autres AWS ressources. Cette mise à jour concerne les applications que vous regroupez et déployez ainsi que les fichiers binaires du serveur de jeu intégrés au SDK du serveur version 5.x ou ultérieure. (Les exécutables du serveur de jeu peuvent déjà demander des informations d'identification à l'aide de l'action `GetFleetRoleCredentials()` 5.x du SDK du serveur.)

Par exemple, si vous souhaitez déployer la version de votre serveur de jeu avec un CloudWatch agent Amazon afin de collecter des métriques d'instance EC2 et d'autres données, l'agent doit être autorisé à interagir avec vos CloudWatch ressources. Pour ce faire, vous devez d'abord configurer un rôle AWS Identity and Access Management IAM () autorisé à utiliser les CloudWatch ressources, puis configurer une flotte avec le rôle IAM et les informations d'identification partagées activés. Lorsqu'Amazon GameLift déploie la version de votre serveur de jeu sur chaque instance EC2, il génère un fichier d'informations d'identification partagé et le stocke sur l'instance. Toutes les applications de l'instance peuvent utiliser les informations d'identification partagées. Amazon actualise GameLift automatiquement les informations d'identification temporaires pendant toute la durée de vie de l'instance.

Vous pouvez activer les informations d'identification partagées lorsque vous créez un parc EC2 géré en utilisant les méthodes suivantes :

- Dans le flux de travail de création de flotte de la GameLift console Amazon.
- Lorsque vous appelez l'opération d'API du GameLift service Amazon `CreateFleet` à l'aide du nouveau paramètre`InstanceRoleCredentialsProvider`.
- Lorsque vous appelez l'opération AWS CLI `aws gamelift create-fleet` avec le paramètre`instance-role-credentials-provider`.

En savoir plus :

- Communiquez avec d'autres AWS ressources de vos flottes, Amazon GameLift Developer Guide
- CreateFleet InstanceRoleCredentialsProvider, Référence d' GameLift API Amazon
- Configurer un rôle de service IAM, Amazon GameLift Developer Guide

28 septembre 2023 : Amazon GameLift lance un nouveau plugin autonome pour Unreal Engine

Versions du SDK mises à jour : GameLift plugin Amazon pour Unreal Engine version 1.0.0

Le GameLift plugin Amazon pour Unreal Engine fournit des outils et des flux de travail qui simplifient les étapes à suivre pour lancer un jeu avec Amazon GameLift pour l'hébergement dans le cloud. Amazon GameLift est un service entièrement géré qui permet aux développeurs de jeux de gérer et de faire évoluer des serveurs de jeux dédiés pour les jeux multijoueurs basés sur des sessions. Le plugin prend en charge les versions UE 5.0, 5.1 et 5.2. Les principales fonctionnalités sont les suivantes :

- Les flux de travail guidés de l'interface utilisateur [dans l'éditeur Unreal] suivent les étapes suivantes :
  - Testez l'intégration de votre jeu avec Amazon GameLift en utilisant votre station de travail locale comme hôte. Ce flux de travail vous permet de configurer une GameLift Anywhere flotte Amazon pour votre machine locale, de lancer des instances de votre serveur de jeu et de votre client, de demander une session de jeu via Amazon GameLift et d'obtenir les informations de connexion pour la nouvelle session de jeu.
  - Déployez une solution d'hébergement cloud Amazon EC2 pour votre serveur de jeu intégré. Ce flux de travail vous aide à configurer votre jeu pour l'hébergement dans le cloud et propose trois scénarios de déploiement différents : déploiement sur une flotte unique, déploiement sur un ensemble de flottes ponctuelles dans plusieurs régions ou déploiement sur un ensemble de flottes avec un FlexMatch entremetteur. La solution pour chaque scénario de déploiement inclut les GameLift ressources Amazon et les AWS ressources de support.
- Possibilité de configurer des profils utilisateur liés à un utilisateur du AWS compte et de définir une AWS région par défaut. Vous pouvez gérer plusieurs profils pour travailler dans différents AWS comptes, utilisateurs de comptes et régions.
- Des fonctionnalités spéciales qui aident à rationaliser les processus GameLift d'intégration et de déploiement d'Amazon, notamment :
  - Chaque solution d'hébergement inclut AWS des ressources de support, notamment un groupe d'utilisateurs Amazon Cognito de base qui fournit des identifiants de joueur uniques, un compartiment Amazon S3 pour le stockage, des notifications d'événements Amazon SNS et des fonctions. AWS Lambda
  - Pour le Anywhere flux de travail, le plugin automatise les paramètres de serveur requis à l'aide d'arguments de ligne de commande.

  - Pour le flux de travail Amazon EC2, chaque solution de déploiement fournit un service client backend intégré utilisant les fonctions Lambda. Le service principal reçoit les demandes des clients du jeu et les transmet au GameLift service Amazon.
- Du contenu pour les tests d'intégration, notamment une carte de jeu de démarrage et deux cartes de test avec des plans de base et des éléments d'interface utilisateur.
- Documentation du plugin avec des conseils d'intégration détaillés et un exemple de code.

Tous les scénarios de déploiement, y compris pour Anywhere et pour les flottes Amazon EC2, utilisent des AWS CloudFormation modèles pour décrire les solutions. Le plugin utilise ces modèles lors du déploiement GameLift des ressources Amazon pour votre jeu. Ces modèles sont inclus dans le téléchargement du GameLift plugin Amazon et sont modifiables. Vous pouvez les utiliser tels quels ou les modifier pour votre jeu.

En savoir plus :

- Intégration de jeux avec le GameLift plugin Amazon pour Unreal Engine, Guide GameLift du développeur Amazon
- Téléchargez le plugin sur GitHub
- À propos de l' GameLift hébergement Amazon
- GameLift Forum Amazon

17 août 2023 : Amazon GameLift propose un hébergement de serveurs de jeux avec des processeurs AWS Graviton

Versions du SDK mises à jour : AWS SDK 1.11.144

Avec Amazon, GameLift vous pouvez désormais héberger vos jeux dans le cloud à l'aide d'instances EC2 équipées de processeurs AWS Graviton. Conçues à l' AWS aide de processeurs basés sur ARM64, les instances Graviton offrent le meilleur rapport prix/performances pour les charges de travail cloud utilisant EC2, avec une amélioration allant jusqu'à 40 % par rapport aux instances x86 comparables. Les derniers processeurs Graviton3 offrent des performances de calcul jusqu'à 25 % supérieures à celles des versions précédentes.

Avec Amazon GameLift, vous pouvez désormais choisir parmi ces nouvelles instances de la famille AWS Graviton :

- Instances basées sur Graviton2 : c6g, c6gn, r6g, m6g, g5g

- Instances basées sur Graviton3 : c7g, r7g, m7g

En savoir plus :

- AWS Processeur Graviton : découvrez les avantages et les utilisations pratiques des instances EC2 basées sur Graviton.
- Premiers pas avec Graviton : obtenez une vue d'ensemble des instances basées sur Graviton et des informations sur la façon dont les applications s'exécutent sur celles-ci en fonction de leur système d'exploitation, de leurs langues et de leur durée d'exécution.

> **Note**
>
> Les instances Graviton Arm nécessitent un GameLift serveur Amazon basé sur le système d'exploitation Linux. Le SDK Server 5.1.1 ou une version ultérieure est requis pour C++ et C#. Le SDK Server 5.0 ou une version ultérieure est requis pour Go. Ces instances ne prennent pas out-of-the-box en charge l'installation de Mono sur Amazon Linux 2023 (AL2023) ou Amazon Linux 2 (AL2).

27 juillet 2023 : Amazon GameLift publie le SDK de serveur 5.1.0 avec un support supplémentaire pour le développement de Unity

Versions du SDK mises à jour : SDK serveur pour C++, C#/Unity, Unreal 5.1.0

La dernière version du SDK Amazon GameLift Server fournit des mises à jour pour C++, C# et le plugin Unreal, ainsi qu'un nouveau plugin à utiliser avec le moteur de jeu Unity. Les développeurs de jeux intègrent le SDK GameLift du serveur Amazon dans les serveurs de jeux qu'ils déploient pour être hébergés sur Amazon GameLift.

La dernière version du SDK du serveur contient les mises à jour suivantes, qui incluent un certain nombre de demandes de clients :

- Télécharger des packages SDK spécifiques à une langue — Le site de GameLift téléchargement Amazon mis à jour contient des packages SDK pour chaque langue. Vous pouvez télécharger les versions actuelles ou antérieures.
- Nouveau plug-in SDK pour serveur C# pour Unity — Le nouveau package SDK pour serveur pour Unity contient des bibliothèques C# intégrées que vous pouvez installer à l'aide du gestionnaire

de packages dans Unity Editor (voir le nouveau guide d'intégration de Unity). Ces bibliothèques incluent les dépendances requises via UnityNuGet. Vous pouvez utiliser ce plugin avec Unity 2020.3 LTS, 2021.3 LTS et 2022.3 LTS pour Windows et Mac OS. Il prend en charge les profils .NET Framework et .NET Standard de Unity, avec .NET Standard 2.1 et .NET 4.x.

- Solution .NET consolidée pour C# — Le SDK du serveur pour C# prend désormais en charge .NET Framework 4.6.2 (mis à niveau depuis 4.6.1) et .NET 6.0 dans une seule solution. .NET Standard 2.1 est disponible avec les bibliothèques créées par Unity.
- Mises à jour 5.1.0 du SDK Server
  - [C++, C#, Unreal] Vous pouvez désormais appeler `InitSDK()` avec ou sans paramètres de serveur. Les serveurs de jeu qui s'exécutent sur des flottes EC2 GameLift gérées par Amazon lisent les paramètres du serveur directement à partir des variables d'environnement. Les serveurs de jeu des GameLift Anywhere flottes Amazon doivent appeler `InitSDK()` avec les paramètres du serveur.
  - [C++, C#, Unreal] Les appels du SDK du serveur ont amélioré les messages d'erreur.
  - [SDK C++] Pour améliorer les délais de création du SDK du serveur, l'indicateur de génération `-DRUN_CLANG_FORMAT` est désactivé par défaut. Vous pouvez l'activer avec `-DRUN_CLANG_FORMAT=1`.
  - [SDK C++] Lorsque vous créez les bibliothèques sans les bibliothèques standard (`-DGAMELIFT_USE_STD=0`), les types de `std::` données `InitSDK()` ne sont plus utilisés.
- Documentation étendue du SDK 5.x pour serveurs
  - Guides de référence du SDK serveur mis à jour pour C++, C#/Unity et Unreal, y compris une couverture étendue de tous les types de données.
    - Référence GameLift du SDK Amazon Server 5.x pour C# et Unity
    - Référence GameLift du SDK Amazon Server 5.x pour C++
    - Référence du SDK GameLift 5.x du serveur Amazon Unreal Engine
  - Nouvelles versions des guides d'intégration du SDK 5 du serveur pour les plugins Unity et Unreal
    - Intégrer Amazon GameLift dans un projet Unity
    - Intégrer Amazon GameLift dans un projet Unreal Engine
- Mises à jour de documentation supplémentaires
  - Documentation révisée pour les opérations de l'API de GameLift service Amazon GetComputeAccessset GetInstanceAccesspour clarifier les procédures d'accès à distance en fonction de la version GameLift du SDK du serveur Amazon utilisée.

- Descriptions révisées GameSessionPlacementpour documenter la façon dont les informations de session de jeu sont transitoires lorsqu'un placement est « en attente ».

13 juillet 2023 : Amazon GameLift ajoute des statistiques sur le matériel de sa flotte

Vous pouvez désormais suivre les indicateurs de performance du matériel pour vos flottes EC2 GameLift gérées par Amazon. Les métriques incluent les métriques d'instance EC2 relatives à l'utilisation du processeur, au volume du trafic réseau et à l'activité de lecture/écriture sur le disque. Pour Amazon GameLift, ces statistiques décrivent toutes les instances actives d'un emplacement de flotte. Vous pouvez consulter ces statistiques matérielles du parc à l'aide d'un CloudWatch tableau de bord Amazon dans le AWS Management Console. Vous pouvez également les consulter sur la GameLift console Amazon dans les détails de la flotte.

En savoir plus :

- Surveillez Amazon GameLift avec Amazon CloudWatch(Mesures pour les flottes), Amazon GameLift Developer Guide

29 juin 2023 : Amazon GameLift lance le support pour Amazon Linux 2023

Versions du SDK mises à jour : AWS SDK 1.11.111

GameLift Les clients d'Amazon peuvent désormais utiliser le système d'exploitation Amazon Linux 2023 pour héberger leurs serveurs de jeux. AL2023 offre plusieurs améliorations par rapport à AL2, notamment en matière de sécurité. Ce système d'exploitation est disponible dans toutes Régions AWS les régions, à l'exception de la Chine.

Les clients peuvent utiliser les nouveaux systèmes d'exploitation Linux et continuer à recevoir des mises à jour de sécurité critiques lorsque le support d'Amazon Linux (AL1) prendra fin en décembre 2023. Support pour Amazon Linux 2 jusqu'en 2025.

En savoir plus :

- FAQ sur le serveur Amazon GameLift Linux
- Comparaison entre Amazon Linux 2 et Amazon Linux 2023
- Liens de référence de GameLift l'API Amazon :
  - AWS Action du SDK `CreateBuild`

- commande CLI `upload-build`
- commande CLI `create-build`

25 mai 2023 : Amazon GameLift FleetIQ ajoute un filtre pour exclure les emplacements de sessions de jeu sur les instances épuisées

Versions du SDK mises à jour : AWS SDK 1.11.87

Si vous utilisez Amazon GameLift FleetIQ pour l'hébergement de jeux, vous pouvez désormais empêcher le placement de sessions de jeu sur des instances actuellement épuisées. Les instances épuisées sont signalées comme devant être arrêtées, mais elles peuvent toujours être sélectionnées pour héberger de nouvelles sessions de jeu si aucune autre ressource d'hébergement n'est disponible. Grâce à cette nouvelle fonctionnalité, vous pouvez totalement exclure l'utilisation d'instances drainantes.

Utilisez cette fonctionnalité lorsque vous appelez `ClaimGameServer` pour trouver des serveurs de jeu disponibles. Ajoutez le nouveau `FilterOption` paramètre et définissez le statut des instances autorisées sur ACTIVE uniquement. En réponse, Amazon GameLift FleetIQ examine uniquement les instances actives lorsqu'il recherche et revendique un serveur de jeu disponible.

En savoir plus :

- ClaimGameServerdans le Amazon GameLift API Reference
- Comment fonctionne FleetIQ dans le guide du développeur Amazon GameLift FleetIQ

16 mai 2023 : Amazon GameLift prend en charge le balisage de répartition des coûts pour les flottes

GameLift Les clients d'Amazon peuvent désormais utiliser des balises de répartition des AWS Billing coûts pour organiser leurs coûts d'hébergement de jeux. Vous pouvez attribuer des balises de répartition des coûts aux ressources individuelles du parc Amazon GameLift EC2 afin de suivre la contribution de vos flottes aux coûts d'hébergement globaux.

En savoir plus :

- Gérez les coûts d'hébergement de vos jeux
- Utilisation des balises de répartition des AWS coûts, guide de AWS Billing l'utilisateur

20 avril 2023 : Amazon GameLift lance le support pour Windows Server 2016

Versions du SDK mises à jour : AWS SDK 1.11.63

GameLift Les clients Amazon peuvent désormais utiliser le système d'exploitation Windows Server 2016 pour héberger leurs serveurs de jeux. Ce système d'exploitation est disponible dans tous les pays Régions AWS. Les clients peuvent utiliser le nouveau système d'exploitation Windows et continuer à recevoir des mises à jour de sécurité critiques alors que Microsoft mettra fin au support de Windows Server 2012 en octobre 2023.

À compter d'aujourd'hui, les nouveaux clients qui ont besoin d'un environnement d'exécution Windows doivent spécifier Windows Server 2016 lors de la création de nouvelles versions de serveurs de jeux destinées à l'hébergement. Les clients existants peuvent continuer à créer de nouvelles versions et flottes avec Windows Server 2012, mais ils doivent terminer la migration avec Windows Server 2016 avant la date de fin du support de Microsoft, le 10 octobre 2023.

Cette mise à jour inclut les modifications de service suivantes :

- Lorsque vous créez une version de serveur de jeu à l'aide des commandes Amazon GameLift SDK ou CLI, vous devez désormais définir explicitement le système d'exploitation. Il n'existe plus de valeur par défaut. Pour déployer votre serveur de jeu sur Windows Server 2016, utilisez la valeur`WINDOWS_2016`.
- Lorsque vous créez une version de serveur de jeu à l'aide de la GameLift console Amazon, vous devez sélectionner un système d'exploitation parmi les valeurs disponibles. Si vous êtes déjà un client disposant de flottes Windows Server 2012 actives, vous pouvez choisir entre l'une `WINDOWS_2012` ou `WINDOWS_2016` l'autre.

En savoir plus :

- Liens de référence de GameLift l'API Amazon :
  - commande CLI `upload-build`
  - commande CLI `create-build`
  - AWS Action du SDK `CreateBuild`
- GameLift FAQ Amazon pour Windows 2012

13 avril 2023 : Amazon GameLift lance le SDK 5.x pour serveur pour Unreal

Versions du SDK mises à jour : Server SDK 5.0.0 pour Unreal

La dernière version du plugin GameLift léger Amazon pour Unreal Engine est désormais basée sur le SDK 5.x GameLift du serveur Amazon. Pour commencer à intégrer votre environnement Unreal Engine à Amazon, GameLift consultez les liens suivants.

En savoir plus :

- Intégrer Amazon GameLift dans un projet Unreal Engine
- Ajoutez Amazon GameLift à votre serveur de jeu
- Référence GameLift du SDK Amazon Server 5.x pour C++

14 mars 2023 : Amazon GameLift lance une nouvelle expérience sur console

La nouvelle GameLift console Amazon inclut les améliorations suivantes :

- Navigation améliorée — Le nouveau volet de navigation facilite la navigation entre les GameLift ressources Amazon.
- Page de GameLift destination Amazon — La nouvelle page de destination fournit des liens vers de la documentation utile, affiche une présentation générale d'Amazon GameLift et fournit une assistance via des liens vers de la documentation, des questions fréquemment posées et AWS re:Post.
- Statistiques Amazon améliorées : CloudWatch les GameLift métriques Amazon sont désormais disponibles à la fois dans la GameLift console Amazon et dans vos CloudWatch tableaux de bord. Cette mise à jour inclut également de nouvelles mesures relatives aux performances, à l'utilisation et aux sessions des joueurs.

En savoir plus :

- Affichage de vos données de jeu sur la console
- Gestion des ressources GameLift d'hébergement Amazon
- Construire un FlexMatch entremetteur

14 février 2023 : Amazon prend GameLift désormais en charge le chiffrement côté serveur pour les rubriques Amazon SNS

Le chiffrement côté serveur (SSE) pour les rubriques SNS chiffre vos données sensibles au repos. SSE utilise des touches AWS Key Management Service (AWS KMS) pour protéger le contenu de vos rubriques SNS.

En savoir plus :

- Configurer une notification d'événement pour le placement des sessions de jeu
- FlexMatchévénements de matchmaking
- Chiffrement au repos

9 février 2023 : le SDK GameLift du serveur Amazon prend en charge .NET 6 avec C #10

Versions du SDK mises à jour : Server SDK 5.0.0 pour .NET 6. Aucune mise à jour du SDK n'est requise.

Si vous utilisez la plateforme de développement en temps réel Unity, continuez à utiliser le SDK 5.0.0 GameLift du serveur Amazon avec .NET 4.6. Unity ne prend pas en charge .NET 6.

En savoir plus :

- Téléchargez la dernière version du SDK du GameLift serveur Amazon sur Amazon GameLift Getting Started
- Référence GameLift du SDK Amazon Server 5.x pour C# et Unity

31 janvier 2023 : le SDK GameLift du serveur Amazon prend en charge le langage Go

Versions du SDK mises à jour : Server SDK 5.0.0 pour Go

En savoir plus :

- Téléchargez la dernière version du SDK du GameLift serveur Amazon sur Amazon GameLift Getting Started
- Référence GameLift du SDK Amazon Server pour Go

1er décembre 2022 : Amazon GameLift lance Amazon GameLift Anywhere et Amazon GameLift Server SDK 5.0

Versions du SDK mises à jour : AWS SDK 1.10.21, SDK serveur 5.0.0 pour C++ et C#

Amazon GameLift Anywhere utilise les ressources de votre serveur de jeu pour héberger les serveurs de GameLift jeux Amazon. Vous pouvez utiliser Amazon GameLift Anywhere pour intégrer vos propres ressources informatiques au calcul EC2 GameLift géré par Amazon afin de répartir vos serveurs de jeu sur plusieurs types de calcul. Vous pouvez également utiliser Amazon GameLift Anywhere pour tester vos serveurs de jeu de manière itérative sans télécharger le build sur Amazon GameLift à chaque itération.

Points forts :

- Nouveaux types de GameLift Anywhere flotte et de calcul Amazon
- Enregistrement des ressources GameLift Anywhere informatiques Amazon
- Cycle d'itération des tests amélioré

Le SDK Amazon GameLift Server 5.0.0 apporte des améliorations au SDK du serveur existant et introduit un nouveau type de ressource, le calcul. Server SDK 5.0.0 prend en charge Amazon GameLift Anywhere et l'utilisation de vos propres ressources informatiques pour l'hébergement de serveurs de jeux.

En savoir plus :

- Référence GameLift du SDK pour serveurs Amazon
- Emplacement de la flotte
- Choisir les ressources GameLift informatiques d'Amazon
- Créez une GameLift Anywhere flotte Amazon

25 août 2022 : Amazon GameLift lance le support pour les Zones Locales

Versions du SDK mises à jour : AWS SDK 1.9.333

Amazon GameLift est désormais disponible dans huit Zones Locales aux États-Unis, ce qui vous permet de déployer vos flottes au plus près des joueurs. Vous pouvez utiliser toutes les GameLift fonctionnalités gérées par Amazon avec les zones locales en ajoutant les zones locales à vos flottes.

Les Zones AWS Locales étendent les ressources et les services jusqu'à la périphérie du cloud, à proximité de grands centres urbains, industriels et informatiques (IT). Cela signifie que vous pouvez déployer des applications qui nécessitent une latence d'une milliseconde à un chiffre plus près des utilisateurs finaux ou des centres de données sur site.

En savoir plus :

- Zones locales
- Emplacement de la flotte
- Créez une flotte GameLift gérée par Amazon

28 juin 2022 : Amazon GameLift lance une nouvelle expérience de console opt-in

La nouvelle GameLift console Amazon inclut les améliorations suivantes :

- Navigation améliorée — Le nouveau volet de navigation facilite la navigation entre les GameLift ressources Amazon.
- Page de GameLift destination Amazon — La nouvelle page de destination fournit des liens vers de la documentation utile, affiche une présentation générale d'Amazon GameLift et fournit une assistance via des liens vers de la documentation, des questions fréquemment posées et AWS re:Post.
- Statistiques Amazon améliorées : CloudWatch les GameLift métriques Amazon sont désormais disponibles à la fois dans la GameLift console Amazon et dans vos CloudWatch tableaux de bord. Cette mise à jour inclut également de nouvelles mesures relatives aux performances, à l'utilisation et aux sessions des joueurs.

En savoir plus :

- Affichage de vos données de jeu sur la console
- Gestion des ressources GameLift d'hébergement Amazon
- Construire un FlexMatch entremetteur

15 février 2022 : FlexMatch ajoute une règle composée et des améliorations supplémentaires

FlexMatch les utilisateurs ont désormais accès aux fonctionnalités suivantes :

- Règle composée — Ajout de la prise en charge des règles de matchmaking composées pour les matchs de 40 joueurs ou moins. Vous pouvez désormais utiliser des instructions logiques pour créer une règle composée afin de former une correspondance. En l'absence de règle composée dans votre ensemble de règles, pour qu'une correspondance soit établie, toutes les règles de l'ensemble de règles doivent être vraies. Avec les règles composées, vous pouvez choisir les règles à appliquer à l'aide des opérateurs logiques suivants : `and`,`or`,`not`, et`xor`.
- Sélection flexible des équipes — Expressions de propriétés de matchmaking mises à jour pour faciliter la sélection d'un sous-ensemble de toutes les équipes disponibles.
- Listes de chaînes plus longues : augmentation du nombre maximum de chaînes de 10 à 100 dans une liste de chaînes contenant les valeurs des attributs des joueurs.

En savoir plus :

- Guide GameLift FlexMatch du développeur Amazon :
  - FlexMatch types de règles
  - FlexMatch expressions de propriété
- AttributeValue: SL

28 octobre 2021 : Amazon GameLift ajoute la prise en charge des flottes multirégionales dans la région Asie-Pacifique (Osaka) ; Amazon FleetIQ ajoute la prise en charge des processeurs GameLift Graviton2 AWS

Versions du SDK mises à jour : AWS SDK 1.9.133

Amazon GameLift est désormais disponible dans la région Asie-Pacifique (Osaka). Les développeurs de jeux peuvent désormais déployer des instances à Osaka à l'aide d' GameLift une flotte multirégionale.

Vous pouvez désormais utiliser les serveurs de jeu hébergés par Graviton2, basés sur l'architecture de processeur ARM, pour obtenir des performances accrues à moindre coût par rapport aux options de calcul équivalentes basées sur Intel.

Points forts :

- Amazon GameLift est désormais disponible dans la région Asie-Pacifique (Osaka).
- Les groupes de serveurs de jeu Amazon GameLift FleetIQ peuvent désormais être configurés pour gérer les familles d'instances Graviton2 c6g, m6g et r6g.

En savoir plus :

- Flotte GameLift multirégionale Amazon
- CreateGameServerGroup
- AWS processeur graviton

20 septembre 2021 : Amazon GameLift lance un plugin pour Unity

Le GameLift plugin Amazon pour Unity version 1.0.0 contient des bibliothèques et une interface utilisateur native qui facilitent l'accès aux GameLift ressources Amazon et l'intégration d'Amazon GameLift dans votre jeu Unity. Vous pouvez utiliser le GameLift plugin Amazon pour Unity afin d'accéder aux GameLift API Amazon et de déployer des AWS CloudFormation modèles pour des scénarios de jeu courants. Le plugin inclut également un exemple de jeu qui fonctionne avec les exemples de scénarios. Vous pouvez utiliser Amazon GameLift Local pour consulter les messages transmis entre le client du jeu et le serveur de jeu afin de découvrir comment un jeu classique interagit avec Amazon GameLift.

Le plugin pour Unity prend en charge Unity 2019.4 LTS et 2020.3 LTS.

Points forts :

- Créez, exécutez et modifiez un exemple de jeu avec différents scénarios, ou créez le vôtre.
- Déployez AWS CloudFormation des exemples de scénarios pour des scénarios de jeu typiques, notamment l'authentification uniquement, une flotte à région unique, des flottes multirégionales avec file d'attente et système de matchmaking personnalisé, des flottes ponctuelles avec file d'attente et un système de matchmaking personnalisé, et. FlexMatch

En savoir plus :

- Intégration de jeux avec le GameLift plugin Amazon pour Unity

30 juin 2021 : FlexMatch ajoute la règle BatchDistance

Vous pouvez utiliser le type de règle BatchDistance pour spécifier une chaîne ou un attribut numérique, ce qui apporte de nombreux avantages à chaque segment.

Points forts :

- Pour les matchs de grande envergure (>40 joueurs), au lieu d'équilibrer les joueurs uniquement en fonction de leurs compétences, vous pouvez désormais obtenir le même équilibre en fonction des compétences, des modes et des cartes. Assurez-vous que tous les participants au match appartiennent à une fourchette de compétences, associez plusieurs attributs numériques tels que la ligue ou le style de jeu, et regroupez en fonction d'attributs de chaîne tels que la carte ou le mode de jeu. Vous pouvez également créer des extensions au fil du temps. Par exemple, vous pouvez créer une extension pour permettre à un plus grand nombre de niveaux de compétence d'entrer dans le match au fur et à mesure que le joueur attend.

  Pour les matchs de moins de 40 joueurs, vous pouvez utiliser une nouvelle expression de règles simplifiée.

3 juin 2021 : mises à jour du SDK client Amazon en GameLift temps réel et du SDK serveur

Versions du SDK mises à jour : SDK client en temps réel 1.2.0, SDK serveur 3.4.0 pour Unreal

Avec cette dernière mise à jour du SDK, vous pouvez désormais intégrer IL2CPP dans vos applications mobiles qui utilisent le SDK client RTS et suivre les meilleures pratiques en matière de frameworks. Vous pouvez également désormais créer le SDK Amazon GameLift Server pour Unreal version 4.26. Cette mise à jour contient des composants qui s'intègrent à votre serveur de jeu Windows ou Linux, notamment les versions C++ et C# du SDK Amazon GameLift Server, Amazon GameLift Local et un plug-in Unreal Engine.

Points forts :

- Ajout de la prise en charge de l'IL2CPP dans le SDK du client RTS et de la création des bibliothèques natives sous forme de frameworks, afin que vous puissiez créer des clients RTS pour les appareils mobiles les plus récents.
- Vous pouvez l'utiliser DescribePlayerSessions() pour obtenir des informations pour une session solo, pour toutes les sessions d'une session de jeu ou pour toutes les sessions de joueur associées à un identifiant solo.
- Vous pouvez l'utiliser GetInstanceCertificate() pour récupérer l'emplacement du fichier d'un certificat TLS codé PEM associé à la flotte et à ses instances.
- Support du SDK serveur créé pour Unreal version 4.26.
- Le SDK C# existant, version 4.0.2, a été vérifié comme compatible avec Unity 2020.3. Aucune mise à jour du SDK n'était requise.

En savoir plus :

- Guide GameLift du développeur Amazon :
  - DescribePlayerSessions()
  - GetInstanceCertificate()

23 mars 2021 : Amazon GameLift ajoute des notifications au placement des sessions de jeu

Versions du SDK mises à jour : AWS SDK 1.8.168

Vous pouvez désormais utiliser les événements pour surveiller l'activité de placement des sessions de jeu dans une file d'attente de session de jeu. Créez une rubrique Amazon Simple Notification Service (Amazon SNS) pour publier des notifications d'événements, ou configurez le suivi des événements à l'aide d'Events. CloudWatch

Points forts :

- Pour chaque file d'attente, vous pouvez définir une chaîne de texte personnalisée à inclure dans tous les messages relatifs aux événements.
- Lorsque vous utilisez une rubrique Amazon SNS, vous pouvez définir des conditions d'accès supplémentaires qui limitent la publication à des files d'attente spécifiques.

En savoir plus :

- Guide GameLift du développeur Amazon :
  - Configurer une notification d'événement pour le placement des sessions de jeu (nouveau)
  - Événements de placement de sessions de jeu (nouveau)
- Référence d'API (AWS SDK)
  - Nouveaux paramètres de file d'attente pour les sessions de jeu `NotificationTarget` et `CustomEventData` : GameSessionQueue, CreateGameSessionQueue, UpdateGameSessionQueue
- GameLiftForum Amazon

16 mars 2021 : Amazon GameLift ajoute des flottes multirégionales et six nouvelles régions

[Versions du SDK mises à jour : AWS SDK 1.8.163]

L'hébergement GameLift géré par Amazon est désormais disponible dans 21 AWS régions. Les nouvelles régions sont Le Cap (`af-south-1`), Bahreïn (`me-south-1`), Hong Kong (`ap-east-1`), Milan (`eu-south-1`), Paris (`eu-west-3`) et Stockholm (`eu-north-1`).

Grâce à la nouvelle fonctionnalité des GameLift flottes multi-sites Amazon, vous pouvez désormais configurer une flotte unique pour héberger vos serveurs de jeu dans l'une ou l'ensemble des 20 régions GameLift prises en charge par Amazon (à l'exception de la région de Pékin). Cette fonctionnalité vise à réduire de manière significative le travail requis pour configurer et gérer les ressources GameLift d'hébergement Amazon dans le monde entier. Des flottes multisites peuvent être créées dans les AWS régions suivantes : `us-east-1` (Virginie du Nord), `us-west-2` (Oregon), `eu-central-1` (Francfort), `eu-west-1` (Irlande), `ap-southeast-2` (Sydney), `ap-northeast-1` (Tokyo) et `ap-northeast-2` (Séoul). Dans toutes les autres régions, vous pouvez continuer à configurer des flottes à site unique selon vos besoins. Toutes les flottes créées avant cette version sont des flottes à site unique. L'utilisation de flottes multisites n'a aucune incidence sur vos coûts d'hébergement. La GameLift tarification d'Amazon est basée sur le type, l'emplacement et le volume d'instances que vous utilisez. (Pour plus d'informations, consultez les GameLifttarifs Amazon.) AWS CloudFormation le support pour les flottes multi-sites sera bientôt disponible.

> **Note**
>
> Les flottes multisites ne sont pas disponibles dans les régions de Chine. Les GameLift ressources Amazon situées dans les régions chinoises ne peuvent pas interagir avec les ressources des autres GameLift régions Amazon ni être utilisées par celles-ci.

Points forts :

- Dans le cas d'un parc multisite, ajoutez explicitement une liste de sites distants. Amazon GameLift déploie des instances du même type et de la même configuration, y compris la configuration de construction et d'exécution, dans la région d'origine de la flotte et sur tous les sites ajoutés.
- Ajustez les paramètres de capacité et la mise à l'échelle pour chaque site indépendamment. Les politiques de dimensionnement automatique s'appliquent à l'ensemble du parc, mais vous pouvez les activer ou les désactiver par emplacement.

- Démarrez de nouvelles sessions de jeu à des emplacements spécifiques de la flotte. Lorsque vous utilisez des files d'attente ou le matchmaking pour placer des sessions de jeu, vous pouvez désormais hiérarchiser le début des nouvelles sessions de jeu en fonction de l'emplacement, du coût d'hébergement et de la latence des joueurs.
- Obtenez des statistiques d'hébergement dans la GameLift console Amazon, agrégées pour tous les sites d'une flotte ou ventilées par emplacement de flotte.

En savoir plus :

- Blog sur les technologies liées aux jeux Amazon
- Référence d'API (AWS SDK)
  - Nouvelles opérations de localisation de flottes : CreateFleetLocationsDescribeFleetLocationAttributes, DescribeFleetLocationCapacity, DescribeFleetLocationUtilization, DeleteFleetLocations
  - Opérations de flotte mises à jour, avec nouveau support multi-sites : CreateFleet,, UpdateFleetCapacity, InstanceLimitsDescribeEC2,,, DescribeInstancesStopFleetActionsStartFleetActions
  - Opérations de placement des sessions de jeu mises à jour, avec une nouvelle priorité et une nouvelle capacité de filtrage : CreateGameSessionQueue, DescribeGameSessionQueues, UpdateGameSessionQueue
  - Opérations de création de sessions de jeu mises à jour, avec une nouvelle prise en charge de la localisation : CreateGameSessionDescribeGameSessions,, DescribeGameSessionDetails, SearchGameSessions
- Guide GameLift du développeur Amazon :
  - Sites GameLift d'hébergement Amazon(mis à jour)
  - Guide GameLift de conception de flotte Amazon (nouveau)

    Élargir la capacité GameLift d'hébergement d'Amazon(mis à jour)
  - Conception d'une file d'attente de sessions de jeu (nouveau)
  - Afficher les détails de la flotte(mis à jour)
- GameLiftForum Amazon

9 février 2021 : Amazon GameLift étend la prise en charge des instances AMD, en mode autonome FlexMatch

Versions du SDK mises à jour : AWS SDK 1.8.139

Cette version inclut les mises à jour suivantes :

- Les groupes de serveurs de jeu Amazon GameLift FleetIQ peuvent désormais être configurés pour gérer les familles d'instances AMD C5a, M5a et R5a. Les types d'instances Amazon EC2 pris en charge, tels que listés pour le GameServerGroup InstanceDefinition, incluent désormais les suivants :
  - 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.12x large, m5a.16x large, m5a.24xlarge
  - r5a.large, r5a.xlarge, r5a.2xlarge, r5a.4xlarge, r5a.8xlarge, r5a.12x large, r5a.16xlarge, r5a.24xlarge

  Remarque : les instances AMD pour FleetIQ ne sont actuellement pas disponibles dans la région de Chine (Pékin). AWS Voir Disponibilité des fonctionnalités et différences de mise en œuvre en Chine.
- L'hébergement de jeux GameLift géré par Amazon prend désormais en charge les instances AMD dans la région de Chine (Pékin), exploitées par Sinnet. Les nouvelles familles d'instances AMD incluent M5a et R5a. Les types d'instances EC2 pris en charge, tels qu'ils sont répertoriés pour le parc InstanceType, incluent désormais les suivants :
  - m5a.large, m5a.xlarge, m5a.2xlarge, m5a.4xlarge, m5a.8xlarge, m5a.12x large, m5a.16x large, m5a.24xlarge
  - r5a.large, r5a.xlarge, r5a.2xlarge, r5a.4xlarge, r5a.8xlarge, r5a.12x large, r5a.16xlarge, r5a.24xlarge
- Amazon GameLift FlexMatch peut désormais être utilisé comme solution de jumelage autonome dans la région de Chine (Pékin), exploitée par Sinnet. Les clients peuvent créer un FlexMatch système de jumelage dans la région de Pékin et configurer le FlexMatchModeparamètre sur STANDALONE. Pour plus d'informations sur FlexMatch l'hébergement GameLift géré par Amazon ou sur une solution d' GameLifthébergement autre qu'Amazon, consultez le manuel du GameLift FlexMatch développeur Amazon.

- Lorsque vous configurez les notifications d'événements pour Amazon GameLift FlexMatch, vous pouvez désormais désigner une rubrique Amazon SNS FIFO comme cible de notification. Pour plus d'informations, consultez :
  - MatchmakingConfiguration NotificationTarget, Référence d' GameLift API Amazon
  - Configurer la notification des FlexMatch événements, Amazon GameLift FlexMatch Developer Guide
  - Présentation d'Amazon SNS FIFO — Messagerie F irst-in-first-out Pub/Sub, blog d'actualités AWS

22 décembre 2020 : le SDK GameLift du serveur Amazon prend en charge Unreal Engine 4.25 et Unity 2020

Versions du SDK mises à jour : Amazon GameLift Server SDK 4.0.2, extension Unreal version 3.3.3

La dernière version du SDK Amazon GameLift Server contient les composants suivants :

- Le plugin Unreal mis à jour a été mis à jour pour être compatible avec Unreal Engine 4.25. L'API n'a pas été modifiée.
- Le SDK C# existant, version 4.0.2, a été vérifié comme compatible avec Unity 2020. Aucune mise à jour du SDK n'était requise.

Téléchargez la dernière version du SDK Amazon GameLift Server sur Amazon GameLift Getting Started.

24 novembre 2020 : Amazon est GameLift FlexMatch désormais disponible pour les jeux hébergés n'importe où

Versions du SDK mises à jour : AWS SDK 1.8.95

Amazon GameLift FlexMatch est un service de jumelage personnalisable pour les jeux multijoueurs. Conçu initialement pour les utilisateurs de l'hébergement GameLift géré par Amazon, il FlexMatch peut désormais être intégré à des jeux utilisant d'autres systèmes d'hébergement peer-to-peer, notamment l'informatique propriétaire sur site et les types primitifs de cloud computing. Les jeux qui utilisent Amazon GameLift FleetIQ pour l'hébergement de jeux sur Amazon EC2 peuvent désormais implémenter le matchmaking avec. FlexMatch

FlexMatch fournit un algorithme de matchmaking robuste et un langage de règles qui vous donnent une grande latitude pour personnaliser le processus de matchmaking afin que les joueurs soient jumelés en fonction des caractéristiques clés des joueurs et de la latence signalée. En outre, FlexMatch propose un flux de travail de demande de matchmaking qui prend en charge des fonctionnalités telles que les parties entre joueurs, l'acceptation des joueurs et le remblayage des matchs. Lorsque vous utilisez FlexMatch un hébergement GameLift géré par Amazon ou des serveurs en temps réel, le système de matchmaking utilise automatiquement Amazon GameLift pour trouver des ressources d'hébergement et démarrer une nouvelle session de jeu pour les matchs nouvellement formés. Lors de l'utilisation en FlexMatch tant que service autonome, le système de matchmaking renvoie les résultats des matchs à votre jeu, qui peut ensuite démarrer une nouvelle session de jeu à l'aide de votre solution d'hébergement.

Les opérations d'API pour FlexMatch font partie de l'API de GameLift service Amazon, qui est incluse dans le AWS SDK et le AWS Command Line Interface (AWS CLI). Cette version inclut ces mises à jour pour prendre en charge le matchmaking autonome :

- La ressource API `MatchmakingConfiguration` présente les modifications suivantes :
  - Nouvelle propriété, `FlexMatchMode` indique si le système de matchmaking est utilisé avec un hébergement GameLift géré par Amazon ou en tant que matchmaking autonome.
  - La propriété n'`GameSessionQueueArns`est pas obligatoire lorsqu'elle `FlexMatchMode` est définie sur autonome.
  - Ces propriétés ne sont pas utilisées avec le matchmaking autonome :`AdditionalPlayerCount`,, `BackfillModeGameProperties`,`GameSessionData`.
- La fonction de remblayage automatique n'est pas disponible avec le matchmaking autonome.

24 novembre 2020 : les instances AMD sont désormais disponibles sur Amazon GameLift

Versions du SDK mises à jour : AWS SDK 1.8.95

La liste des types d'instances Amazon EC2 pris en charge par Amazon inclut GameLift désormais trois nouvelles familles d'instances : C5a, M5a et R5a. Ces familles se composent d'instances optimisées pour le calcul AMD qui sont alimentées par des processeurs AMD EPYC fonctionnant à des fréquences allant jusqu'à 3,3. GHz. Les instances AMD sont compatibles x86 ; les jeux actuellement exécutés sur Amazon GameLift peuvent être déployés sur des types d'instances AMD sans modification. Les nouvelles instances sont disponibles dans les AWS régions suivantes :

USA Est (Virginie du Nord et Ohio), USA Ouest (Oregon et Californie du Nord), centre du Canada (Montréal), Amérique du Sud (Sao Paulo), Europe centrale (Francfort), UE Ouest (Londres et Irlande), Asie-Pacifique sud (Mumbai), Asie-Pacifique nord-est (Séoul et Tokyo) et Asie-Pacifique sud-est (Singapour et Sydney).

Les nouvelles instances AMD incluent :

- 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.12x large, m5a.16x large, m5a.24xlarge
- r5a.large, r5a.xlarge, r5a.2xlarge, r5a.4xlarge, r5a.8xlarge, r5a.12x large, r5a.16xlarge, r5a.24xlarge

En savoir plus :

- Blog sur les technologies liées aux jeux Amazon
- Tarification des GameLift instances Amazon
- Instances Amazon EC2 dotées de processeurs AMD EPYC
- GameLiftForum Amazon

11 novembre 2020 : mise à jour de version GameLift du SDK Amazon Server

Versions du SDK mises à jour : Amazon GameLift Server SDK 4.0.2

La nouvelle version 4.0.2 du SDK pour serveurs corrige un problème connu lié au fonctionnement de l'API. `StartMatchBackfill()` Cette opération renvoie désormais une réponse correcte à une demande de remplissage correspondant.

Le problème n'a pas affecté le processus de remblayage des matchs, et le fonctionnement de cette fonctionnalité n'a pas changé. Le problème a peut-être eu un impact sur la messagerie du journal et la gestion des erreurs pour les demandes de remplacement des matchs.

Téléchargez la dernière version du SDK Amazon GameLift Server sur Amazon GameLift Getting Started.

5 novembre 2020 : Nouvelles personnalisations de FlexMatch l'algorithme

FlexMatch les utilisateurs peuvent désormais ajuster les comportements par défaut suivants pour le processus de matchmaking. Ces personnalisations sont définies dans un ensemble de règles de matchmaking. Aucune modification n'a été apportée aux GameLift SDK Amazon.

- Prioriser les tickets de remblayage : Vous pouvez choisir d'augmenter ou de diminuer la priorité des tickets de remblayage lorsque vous recherchez des correspondances acceptables. La priorisation des tickets de remblayage est utile lorsque la fonction de remplissage automatique est activée. Utilisez la propriété de l'algorithme`backfillPriority`.
- Pré-tri pour optimiser la cohérence et l'efficacité des matchs : configurez votre système de jumelage pour qu'il prétrie le pool de tickets avant de regrouper les tickets pour évaluation. En triant les tickets en fonction des attributs clés des joueurs, les matchs qui en résultent ont tendance à avoir des joueurs plus similaires en ce qui concerne ces attributs. Vous pouvez également améliorer l'efficacité du processus d'évaluation en effectuant un tri préalable sur les mêmes attributs que ceux utilisés dans les règles de match. Utilisez la propriété de l'algorithme `sortByAttributes` avec la `strategy` propriété définie sur « sorted ».
- Ajustez la façon dont les temps d'attente pour les extensions sont déclenchés : Choisissez entre le déclenchement des extensions en fonction de l'âge du ticket le plus récent (par défaut) ou du ticket le plus ancien dans un match incomplet. Le déclenchement sur le ticket le plus ancien a tendance à terminer les matchs plus rapidement, tandis que le déclenchement sur le ticket le plus récent améliore la qualité des matchs. Utilisez la propriété de l'algorithme`expansionAgeSelection`.

17 septembre 2020 : Amazon GameLift met à jour le SDK du serveur

Versions du SDK mises à jour : Amazon GameLift Server SDK 4.0.1

Le nouveau SDK pour serveurs contient les mises à jour suivantes :

- API C# version 4.0.1
  - Le fonctionnement de l'API n'TerminateGameSession()est plus pris en charge. Remplacez par un appel à pour mettre fin ProcessEnding() à la fois à une session de jeu et au processus du serveur.
  - Un problème connu lié à l'opération GetInstanceCertificate() est résolu.
  - L'opération renvoie GetTerminationTime() désormais une valeur de type de données AwsDateTimeOutcome.
- API C++ version 3.4.1

  - L'opération n'TerminateGameSession()est plus prise en charge. Remplacez-le par un appel à pour mettre fin ProcessEnding() à la fois à une session de jeu et au processus du serveur.
- Plug-in Unreal Engine version 3.3.2
  - L'opération n'TerminateGameSession()est plus prise en charge. Remplacez-le par un appel à pour mettre fin ProcessEnding() à la fois à une session de jeu et au processus du serveur.
  - L'opération de rappel `OnUpdateGameSession` est ajoutée pour prendre en charge F ProcessParameters le remplissage des matchs.

Téléchargez la dernière version du SDK Amazon GameLift Server sur Amazon GameLift Getting Started.

27 août 2020 : Amazon GameLift FleetIQ pour l'hébergement de jeux avec Amazon EC2 (disponibilité générale)

Versions du SDK mises à jour : AWS SDK 1.8.36

La solution Amazon GameLift FleetIQ pour l'hébergement de jeux à faible coût dans le cloud sur Amazon EC2 est désormais disponible pour tous. Amazon GameLift FleetIQ permet aux développeurs d'héberger des serveurs de jeux directement sur des instances Amazon EC2 Spot en optimisant leur viabilité pour l'hébergement de jeux. Les développeurs de jeux peuvent utiliser Amazon GameLift FleetIQ avec de nouveaux jeux ou pour augmenter la capacité des jeux existants. Cette solution prend en charge l'utilisation de conteneurs ou d'autres AWS services tels que AWS Shield et Amazon Elastic Container Service (Amazon ECS).

Cette version de disponibilité générale inclut les mises à jour suivantes de la solution Amazon GameLift FleetIQ :

- Une nouvelle opération d'API `DescribeGameServerInstances` renvoie des informations, y compris le statut, sur toutes les instances actives d'un groupe de serveurs de GameLift jeux Amazon FleetIQ.
- La nouvelle stratégie d'équilibrage configure un groupe de serveurs de jeu pour utiliser uniquement les instances à la demande. `ON_DEMAND_ONLY` Vous pouvez mettre à jour la stratégie d'équilibrage d'un groupe de serveurs de jeu à tout moment, ce qui permet de basculer entre l'utilisation d'instances ponctuelles et d'instances à la demande selon les besoins.
- Les éléments d'aperçu suivants ont été supprimés pour des raisons de disponibilité générale :
  - Utilisation de clés de tri personnalisées pour les ressources du serveur de jeu. Les serveurs de jeu peuvent être triés en fonction de l'horodatage des inscriptions.

- Balisage des ressources du serveur de jeu.

16 avril 2020 : Amazon GameLift met à jour le SDK du serveur pour Unity et Unreal Engine

Versions du SDK mises à jour : Amazon GameLift Server SDK 4.0.0, Amazon Local 1.0.5 GameLift

La dernière version du SDK Amazon GameLift Server contient les composants mis à jour suivants :

- Version 4.0.0 du SDK C# mise à jour pour Unity 2019.
- La version 3.3.1 du plugin Unreal a été mise à jour pour les versions 4.22, 4.23 et 4.24 d'Unreal Engine.
- Amazon GameLift Local version 1.0.5 a été mis à jour pour tester les intégrations utilisant le SDK du serveur C# version 4.0.0.

Téléchargez la dernière version du SDK Amazon GameLift Server sur Amazon GameLift Getting Started.

2 avril 2020 : Amazon GameLift FleetIQ est disponible pour l'hébergement de jeux sur EC2 (version préliminaire publique)

Versions du SDK mises à jour : AWS SDK 1.7.310

La fonctionnalité Amazon GameLift FleetIQ optimise la viabilité des instances Spot à faible coût destinées à l'hébergement de jeux. Cette fonctionnalité est désormais étendue aux clients qui souhaitent gérer leurs ressources d'hébergement directement plutôt que par le biais du GameLift service géré Amazon. Cette solution prend en charge l'utilisation de conteneurs ou d'autres AWS services tels que AWS Shield et Amazon Elastic Container Service (Amazon ECS).

En savoir plus :

GameTech article de blog sur Amazon GameLift FleetIQ

19 décembre 2019 : amélioration de la gestion des AWS GameLift ressources pour Amazon

Versions du SDK mises à jour : AWS SDK 1.7.249

Vous pouvez désormais tirer parti des outils de gestion des AWS ressources avec Amazon GameLift Resources. En particulier, toutes les GameLift ressources Amazon clés (builds, scripts, flottes, files d'attente de sessions de jeu, configurations de matchmaking et ensembles de règles de matchmaking) se voient désormais attribuer des valeurs Amazon Resource Name (ARN). Un ARN de ressource fournit un identifiant cohérent unique dans toutes les AWS régions. Ils peuvent être utilisés pour créer des politiques d'autorisations spécifiques aux ressources AWS Identity and Access Management (IAM). Les ressources se voient désormais attribuer un ARN ainsi que l'identifiant de ressource préexistant, qui n'est pas spécifique à une région.

En outre, les GameLift ressources Amazon prennent désormais en charge le balisage. Vous pouvez utiliser des balises pour organiser les ressources, créer des politiques d'autorisation IAM pour gérer l'accès à des groupes de ressources, personnaliser la ventilation des AWS coûts, etc. Lorsque vous gérez les balises pour les GameLift ressources Amazon, utilisez les actions de GameLift l'API Amazon `TagResource()``UntagResource()`, et`ListTagsForResource()`.

En savoir plus :

- TagResourcedans le Amazon GameLift API Reference
- Balisage des ressources AWS dans les Références générales AWS
- Noms des ressources Amazon dans la référence AWS générale

14 novembre 2019 : Nouveaux AWS CloudFormation modèles, mises à jour dans la région de Chine (Pékin)

Versions du SDK mises à jour : AWS SDK 1.7.210

AWS CloudFormation modèles pour Amazon GameLift

Les GameLift ressources Amazon peuvent désormais être créées et gérées via AWS CloudFormation. Les modèles de AWS CloudFormation construction et de flotte existants ont été mis à jour pour s'aligner sur les ressources actuelles, et de nouveaux modèles sont désormais disponibles pour les scripts, les files d'attente, les configurations de matchmaking et les ensembles de règles de matchmaking. AWS CloudFormation les modèles simplifient considérablement la gestion de groupes de AWS ressources connexes, en particulier lors du déploiement de jeux dans plusieurs régions.

En savoir plus :

- Référence au type de GameLift ressource Amazon dans le guide de AWS CloudFormation l'utilisateur
- Gérez les ressources à l'aide AWS CloudFormationdans le guide du GameLift développeur Amazon

Glossaire AWS

Pour connaître la terminologie la plus récente d'AWS, consultez le Glossaire AWS dans la Référence Glossaire AWS.

Les traductions sont fournies par des outils de traduction automatique. En cas de conflit entre le contenu d'une traduction et celui de la version originale en anglais, la version anglaise prévaudra.